Least Squares Ranking on Graphs*

Anil N. Hirani†[1], Kaushik Kalyanaraman[1], and Seth Watts[2]

[1]Department of Computer Science, University of Illinois at Urbana-Champaign
[2]Department of Mech. Sci. & Eng., University of Illinois at Urbana-Champaign

Abstract

Given a set of alternatives to be ranked, and some pairwise comparison data, ranking is a least squares computation on a graph. The vertices are the alternatives, and the edge values comprise the comparison data. The basic idea is very simple and old – come up with values on vertices such that their differences match the given edge data. Since an exact match will usually be impossible, one settles for matching in a least squares sense. This formulation was first described by Leake in 1976 for ranking football teams and appears as an example in Professor Gilbert Strang's classic linear algebra textbook. If one is willing to look into the residual a little further, then the problem really comes alive, as shown effectively by the remarkable recent paper of Jiang et al. With or without this twist, the humble least squares problem on graphs has far-reaching connections with many current areas of research. These connections are to *theoretical computer science* (spectral graph theory, and multilevel methods for graph Laplacian systems); *numerical analysis* (algebraic multigrid, and finite element exterior calculus); other *mathematics* (Hodge decomposition, and random clique complexes); and *applications* (arbitrage, and ranking of sports teams). Not all of these connections are explored in this paper, but many are. The underlying ideas are easy to explain, requiring only the four fundamental subspaces from elementary linear algebra. One of our aims is to explain these basic ideas and connections, to get researchers in many fields interested in this topic. Another aim is to use our numerical experiments for guidance on selecting methods and exposing the need for further development. Many classic Krylov iterative methods worked well for small to moderate-sized problems, with trade-offs described in the paper. Algebraic multigrid on the other hand was generally not competitive on these graph problems, even without counting the setup costs.

Keywords: Exterior calculus; Hodge theory; Laplace-deRham operators; Graph Laplacian; Algebraic multigrid; Krylov methods; Poisson's equation; Chain complex; Cochain complex; Random clique complex

MSC Classes: 65F10, 65F20, 58A14, 68T05, 05C50; **ACM Classes:** G.1.3, F.2.1, G.2.2

1 Introduction

This paper is about ranking of items, of which some pairs have been compared. The formulation we use (and which we did not invent) leads to a least squares computation on graphs, and a deeper analysis requires a second least squares solution. The topology of the graph plays a role, in a way that will be made precise later. The usual graph Laplacian plays a central role in the first least squares problem. But the key actor in the second problem is another Laplacian, hardly studied in theoretical computer science, but well-studied in numerical analysis.

*A preliminary version of this paper was originally posted on arXiv as "Least Squares Ranking on Graphs, Hodge Laplacians, Time Optimality, and Iterative Methods". The present version is much expanded in scope and includes many new numerical experiments.

†Author for correspondence: hirani@cs.illinois.edu; http://www.cs.illinois.edu/hirani

The formulation as two least squares problems is akin to finding the gradient part of a vector field and then finding the curl part. That in turn, is related to solving an elliptic partial differential equation. The setting for the ranking problem however, is obviously different from that for vector fields and differential equations. Instead of domains that are approximated by meshes, one has *general* graphs as a starting point. We try to convey the conceptual and algorithmic implications of these connections and differences. Another, more practical motivation, is to give guidance on which numerical methods to use for the ranking problem. As a side benefit, we are able to point out some directions in numerical linear algebra and other areas that should be developed further.

1.1 Ranking and pairwise comparisons

Even without the many connections brought out in this paper, ranking is an important problem. Human society seems to have a preoccupation with ranking. Almost nothing is immune from society's urge to rank and use rankings. Rankings are used for marketing, for decision-making, for allocation of resources, or just for boasting or urging improvements. Some rankings are based on opinions of a person. But some are based on opinion polls or some other type of numerical data, allowing the use of simple statistics and computational tools. Ranking based on data often consists of computing some weighted average quantity for each item to be ranked. Then sorting on this number yields the ranking. However, the ranking problem has more interesting formulations and algorithms when comparisons and links between the objects are part of the data. A well-known example is the PageRank algorithm used by Google for ranking webpages, which uses the linked structure of the web (in addition to auctioned keywords) to present webpages ranked by "importance" [50].

Our focus is on *pairwise comparisons*. As such, this is a very different starting point than PageRank and very different from ranking each item independently. Our interest in pairwise comparisons was initiated by the recent remarkable paper of Jiang et al. [35]. By using exterior calculus concepts and Hodge decomposition, they motivated the ranking problem in a very different context and manner than what had been done before. (The less common of these terms will be described later. One of the pedagogical advantages of the ranking problem is the ease with which this can be done using graphs.)

The exterior calculus approach of Jiang et al. resonated very deeply with us. We have been working with discretizations of exterior calculus for a long time, albeit never with a mind towards ranking [34]. Exterior calculus is the language of modern physics [1, 27] and can be thought of as the generalization of vector calculus to smooth curved domains (manifolds). The discretizations of this calculus have made inroads into computer graphics [22], and numerical partial differential equations [4] with great effect. Thanks to the work of Jiang et al. we hope for a similar development in graph computations, especially in the case of ranking. This is the development that we are trying to urge forward in this paper.

1.2 Contributions and goals of this paper

In this paper you will find the important concept of Hodge decomposition reduced to the basics of linear algebra, beyond which it probably cannot further be simplified. You will also find many numerical linear algebra approaches for least squares ranking on graphs, suitable for serial computer implementation. There are many numerical experiments, some with expected results, and some with suggestive results, but most have never appeared in literature before. Many of the results of these experiments call out for further investigations in a variety of fields which we point out along the way. The subject matter is new and timely, and attention needs to be drawn to it, even if there are no deep theorems yet in the numerical analysis aspects of the problem. We hope that the reader will appreciate the novelty of the experiments and their broad implications that we point out. The simplicity of the exposition is aimed at wide dissemination of the ideas. Our main aim is to generate enthusiasm and further work in the

computational mathematics community, in this broad arena for numerical problems on graphs.

Use of orientation instead of skew symmetry: As a point of departure from previous work, we have a very different implementation than what is implied by Jiang et al. This difference stems from our experience in discrete and finite element exterior calculus. Jiang et al. rely on skew-symmetric tensors as the basic objects of their formulation, whereas we capture the skew-symmetry using a vector of values associated with oriented objects. This results in savings in space and time, and provides a simpler formulation which is easy to implement. This is an example of the synergy of studying discrete calculus on meshes and graphs in a common framework. This synergy is also on display in a recent book by Grady and Polimeni [31]. Their chapter on ranking does reference Jiang et al. but does not discuss the ideas of that paper. In contrast to their book, our paper is concerned solely with the problem of least squares ranking on graphs.

Guidance on solvers: Our focus on a single fundamental problem allows us to probe the numerical linear algebra aspects much more deeply. We compare iterative Krylov methods [56, 62] and algebraic multigrid [14, 55], and point out some problems ripe for further development. Specifically, algebraic multigrid is not competitive for these graph problems. The recent multilevel solver of Koutis el al. [40] should also be useful for the ranking problem. We were not able to experiment with it. We could not find a reliable complete implementation at the time of writing this paper and could not get our implementation to work. The authors of that solver should be able to modify our code easily, to compare the performance of their solver with Krylov and multigrid methods. The solver of Koutis et al. is designed for graph Laplacians and so it should be quite well suited for the first least squares problem, as we will reason later. We will see that the second least squares problem involves a different type of Laplacian and the solver of Koutis et al. will likely *not* have any special advantage when used for such problems.

Proposal for a spectral simplicial theory: As mentioned earlier, the least squares problems for ranking on graphs involve two types of Laplacians. The eigenvalues and eigenvectors of these operators are worthy of study. Laplacians on manifolds have been studied extensively in geometry [18]. However, the study of Laplacian spectra on graphs is more recent. The first least squares problem involves the graph Laplacian, which is a very well-studied operator. It is the spectrum of this graph Laplacian (and closely related operators) that is studied in the very useful and beautiful spectral *graph* theory [21]. We are proposing that a similar spectral *simplicial* theory should be developed for simplicial complexes. An example application is that the matrix spectrum is useful for understanding the performance of iterative solvers. We hope however, that the payoff from a spectral simplicial theory will go well beyond that. A glimpse of this potential is in the topic of the next paragraph. In that application the eigenvectors of the zero eigenvalue play a role as will be described in detail later.

Numerical experiments on topology of clique complexes: One can use the numerical linear algebra approach embodied in our paper to conduct experiments in the new field that studies the topology of random clique complexes [37]. A clique complex is a graph augmented by its complete subgraphs which are considered as additional structures on the graph. This field generalizes the established field of random graph theory [11, 25]. The concepts of connectivity of a graph generalize naturally to questions about the homology of clique complexes. We include several examples in which numerical techniques for ranking on graphs lead to experiments on clique complexes. In one of these examples we reproduce some recent theoretical results for clique complexes of random graphs [37]. In fact, our experiments on random graphs are also suggestive of new conjectures and refinements of the existing theorems. We also include some experiments on the 1-cohomology of scale-free graphs [5], which is something that has not been analyzed theoretically elsewhere.

Suggestions for Graph 500 benchmarks: In the high performance computing community, a move is afoot to establish and maintain a Graph 500 list. This is like the Top 500 list of supercomputers that is

regularly updated, but focused on graph problems. Benchmark problems are being developed in the process. We want to draw attention to least squares ranking on graphs as a source of benchmarks. In the field of high performance computing, problems like least squares and linear systems for elliptic partial differential equations have been an important source of problems. These have led to many developments, such as in domain decomposition, preconditioners, and iterative methods. The problem of least squares ranking on graphs also involves Laplacians, but these are graph Laplacians and other Laplacians on *general* graphs. When very large ranking problems on diverse architectures are attempted, it is likely that new developments will be needed. At the same time, the problems are easy to set up and some old codes from differential equations can be used right away. Thus the least squares ranking on graphs is a good crossover problem and a bridge from Top 500 to Graph 500.

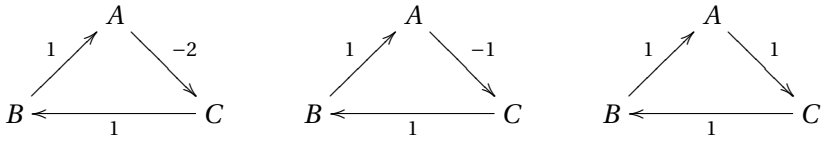
2 Least Squares and Ranking

Let us first examine more carefully why two least squares problems are involved in this formulation of ranking. Given is a set of items to be ranked and some real-valued pairwise comparisons. Not all pairs need to have been compared. Each given pairwise comparison represents how much one alternative in a pair is preferred over the other.

The data can be represented as a weighted directed graph, where each comparison between a pair of items is represented by two edges of opposite direction and equal weight between the items (this is not the formulation we use). This leads to the skew-symmetric 2-tensor representation of comparisons used by Jiang et al. However, this is equivalent to a simple, weighted, undirected graph whose edges are oriented. But this is exactly an oriented abstract simplicial 1-complex, which we will define precisely in Section 4. The edge orientations are arbitrary, and the pairwise score simply changes sign if the opposite edge orientation is used. Without loss of generality, we will usually only consider connected graphs. (Multiple component graphs result in independent ranking problems, one for each component.)

One version of the ranking problem is to find real-valued scores for the vertices, which implies their global rank order, such that the values represent the strength of the rank. The task translates to finding vertex values whose differences are the edge values. However, it is not always possible to compute this exactly. Every loop in the graph has the potential to make existence of such vertex values impossible if the edge values, taken with signs, do not add up to zero as the loop is traversed. What saves this procedure is that, the closest possible global ranking is the vertex value assignment whose differences reproduce the pairwise edge data in the least squares sense. This is a very simple and an old idea that was used for ranking football teams by Leake [42]. The residual, i.e., the part of the edge data that could not be matched, represents inconsistencies in the pairwise data.

Remark 2.1. The term *consistent* is used in a technical sense above. The edge data is consistent if the sum of edge weights (taking orientations into account) around every loop is zero. Consistency is equivalent to having a zero residual in the least squares sense. Least squares ranking can still be computed for inconsistent edge data. For example, the data illustrated in the left graph below is consistent while the other two are not. However, a least squares ranking is possible on each. In particular, A will be the winner and C the loser in the first two cases, and there will be a three-way tie in the last one.



A recent extension of Leake's idea was given by Jiang et al. [35] who examine the residual, decomposing it into local and global inconsistencies, using a second least squares formulation. This time a

3-tensor, i.e., a 3-dimensional matrix, is involved as an unknown in their formulation. As we will show, equivalent least squares problems can be formulated using vectors rather than matrices or 3-tensors to represent the data and unknowns.

We will however follow the point-of-view introduced by Jiang et al., who posed the ranking problem as a discrete *Hodge decomposition* of the pairwise data treated as a *cochain* on a *simplicial 2-complex*. After a discussion of ranking methods in Section 3, in Sections 4 and 5 we will define the terms emphasized in the previous sentence. This will help us clarify the connection to vector field decomposition, elliptic partial differential equations, and topology of complexes.

We emphasize that Leake's idea was to use least squares to find values for each vertex in the graph, given (generally inconsistent) edge data. This is very different from the use of least squares to fit data to a model equation in the sense of statistical regression. It is better to think of the methods of Leake and Jiang et al. as being projections or decompositions, which is the viewpoint that we will take in this paper.

3 Other Ranking Methods

The problem of ranking has been addressed in many areas such as social choice theory, sports analysis, ranking of web pages and machine learning. In this section, we compare and contrast many of these methods with least squares ranking on graphs.

3.1 Social choice theory

In social choice theory, the goal is to rank alternatives based on preferences of a number of voters. Often, the methods are concerned with *ordering* alternatives but not quantification of their relative strengths. A common assumption is that voters indicate their preferences for all alternatives [35]. This results in a complete graph for each voter with only edge orientations specified. However, these may not be reasonable for many real data sets such as internet and e-commerce applications, and ranking of sports teams.

In Condorcet's method [12], a winning (or highest ranked) alternative is one which is preferred over every other in pairwise comparisons. If no such alternative exists, some tie-breaking rule is required . This is equivalent to selecting a winner based on edge orientations of the ranking graph alone and not their weights. On the other hand, least squares ranking uses both the direction and weight of edges to rank all alternatives. The Borda count [12] method ranks alternatives based on a weighted vote scheme. In contrast to least squares ranking, it does not use pairwise comparisons. The Kemeny rule [64] works by testing all $n!$ possible orderings of n alternatives against input edge orientations and weights to arrive at a ranking. Consequently, it is NP-hard [6] while least squares ranking is polynomial time and additionally provides relative strengths of alternatives. Tideman ranking [59] generates a connected, acyclic graph by preferentially retaining edges with highest weights, which do not form a cycle, circumventing inconsistency. In graphs which are originally connected and acyclic, least squares ranking produces the same results as Tideman ranking. However, for general graphs, least squares ranking accounts for inconsistency. Tideman ranking is also ambiguous when all edges have equal weights resulting in a ranking that depends on which edges are removed.

3.2 Sports analysis

Ranking of sports teams is another rich source for ranking methodologies. In addition to satisfying requirements of fairness, rankings are often used to also predict future outcomes amongst teams, which requires some measure of "accuracy" of the rankings. (A method which consistently ranks losing teams highly is not accurate.) Elo ranking [24, 30] treats each alternative as a random variable with a given

mean and variance. When two alternatives are compared (e.g., two players in a game of chess) their strengths and variances are updated based on a set of rules. The rules take into account both the outcome and the strength of the alternatives before the comparison. A global ranking is then a sorting of their mean strengths. Thus, the Elo ranking works exclusively via local interactions whereas least squares ranking always results in a globally optimal consistent ranking. Random walk methods [15, 16, 41] rank alternatives by finding the steady state of a Markov chain. The Markov chain models voter opinions on alternatives when pairwise comparisons and rules for changing votes are given. There may be some connection between these methods and least squares ranking given the connections between random walks and solving Laplace's equation [54]. Keener [38] proposed ranking alternatives such that their ranks were proportional to their strengths, as defined by a preference matrix which records the results of pairwise comparisons. The rankings are found as the Perron-Frobenius eigenvector of the preference matrix. (Subsequent models also defined the ranks as the fixed point of a nonlinear strength function and using maximum likelihood methods amongst others.) Least squares ranking finds ranks which optimally account for the input data, rather than requiring proportionality. Additionally, it is easier to recompute ranking on a given graph for new data by solving the least squares equations with a new right hand side. In contrast, Keener's method requires creating a new preference matrix and finding its Perron-Frobenius eigenvector. Finally, for this eigenvector to exist, the preference matrix must be irreducible which constrains admissible data on the graph but there is no such requirement for least squares ranking. Chartier et al. [17] analyze several popular ranking methods to determine their stability and sensitivity to perturbations in their input data. The perturbations are taken about a perfect season, which is equivalent to ranking on a complete graph with consistent edge data. Of particular interest is the analysis of the Massey [44] ranking method, which is the first least least squares system (15), which shows that the ordinal ranks provided by least squares ranking are quite stable in the presence of perturbations in the input data.

3.3 Other ranking methods

PageRank [50] and HITS [39] are popular methods for ranking web pages based on hyperlinks between them. These are different from the pairwise ranking formulation we study, where there is a value given for how much one alternative is preferred over the other. Ranking techniques in machine learning, like in social choice theory, tend to be concerned with ordering alternatives without exploring their relative strengths. Also most such algorithms are for supervised learning (learning using training data). For example, the problem of learning to rank on a graph addressed by Agarwal [2] is a supervised learning problem. Moreover, it is formulated as a constrained quadratic programming problem. The constraint function in their case is similar to the least squares ranking objective function. The matrix completion method of Gleich and Lim [29] requires a matrix singular value decomposition at each iteration, which is more expensive than least squares ranking using sparse iterative methods.

4 Preliminaries

We will see that a convenient language for revealing the connection of ranking with other fields consists of very basic notions of exterior calculus and cell complexes, which we will quickly recall in this section. This is also needed to frame the two least squares problems as Hodge decomposition, which we do in Section 6. For the first least squares problem we need only the graph described in Section 2. But for the second least squares problem of ranking we will need to use basic ideas about cell complexes and functions on them. We review the irreducible minimum of the basic terminology and concepts that we need from algebraic topology and exterior calculus in this section. For more details on algebraic topology see [48] and for exterior calculus see [1].

4.1 From graphs to complexes

An *abstract simplicial complex* K is a collection of finite non-empty sets called *simplices* such that if a simplex σ is in K then so is every non-empty subset of σ [48]. The elements of a simplex are called its *vertices*. A simplex with $p+1$ vertices is said to have *dimension* p and is referred to as a p*-simplex*. The dimension of a complex is the dimension of the highest dimensional simplex in it. We will refer to a p-dimensional abstract simplicial complex as a p-complex. An *orientation* of a simplex is an equivalence class of permutations (orderings) of its vertices. All even permutations fall into one class and the odd ones into the other class. Thus all simplices of dimension 1 or more have two possible orientations while a vertex has only one orientation. An *oriented* simplex is a simplex along with a choice of an orientation for it. An oriented abstract simplicial complex is one in which all simplices have been oriented. For the applications considered in this paper, the orientations are arbitrary.

Let G be an oriented weighted simple graph, i.e., the edges have been oriented arbitrarily. Then G is an abstract simplicial 1-complex. The vertices of G are the 0-simplices and the edges are the 1-simplices. A p*-clique* of G is a complete subgraph with p vertices. The graph G can be augmented by cliques to make it an abstract simplicial complex of higher dimension. In particular, augmenting G by including the p-cliques, for all $3 \leq p \leq d+1$ yields a d-dimensional simplicial complex which we will refer to as the d-dimensional *clique complex* of G. For the first least squares problem of ranking we only need G to be a graph. For the second problem we need the 2-dimensional clique complex in which the 3-cliques (i.e., triangles) have been oriented arbitrarily. Thus we will augment G by including the triangles of the graph and we will refer to this augmented structure also as G.

Remark 4.1. The 3-cliques are loops of length three. All the results of this paper are valid if we include loops of length ≥ 3 up to some finite length. This yields not a 2-dimensional *simplicial* complex but a 2-dimensional *cell* complex [48]. In the rest of this paper, the reader may substitute the word "cell" for "simplex" or "simplicial" without changing the results.

4.2 Chains and cochains

Let $C_p(G;\mathbb{R})$ be the space of real-valued functions on the oriented p-simplices of G, such that the function changes sign on a simplex when its orientation is flipped. (In algebraic topology, usually one starts with integer valued chains $C_p(G;\mathbb{Z})$ [48].) These functions are called real-valued p*-chains*, or p*-dimensional* chains. Since they take values in reals, the space of p-chains forms a vector space. We will use $C_p(G)$ to abbreviate $C_p(G;\mathbb{R})$. The *elementary chain basis* for $C_p(G)$ consists of functions that are 1 on a particular p-simplex and 0 on the rest. Thus the dimension $\dim C_p(G)$ is the number of p-simplices in G, which we will refer to by the symbol N_p.

The numerical data in the ranking problem are best viewed as real-valued linear *functionals* on the spaces of chains. The reason for using the space of functionals rather than the chains themselves will become clear when we make the analogy with vector calculus in Section 4.6. The spaces of functionals are the vector space duals of $C_p(G)$ and are denoted $C^p(G;\mathbb{R})$, or $C^p(G)$ and called the spaces of p*-cochains*. (Note that cochain spaces have indices on top.) The *elementary cochain basis* for $C^p(G)$ consists of the cochains that are 1 on an elementary chain and 0 for the other elementary chains, i.e., it is the basis dual to the elementary chain basis, where the duality is in the sense of vector space duality.

4.3 Boundary and coboundary operators

At first sight, the extra structure of chains and cochains of the previous subsection seems like extra baggage in the ranking problem. However, the chains and cochains come with the boundary and coboundary operators that we will now recall. These provide the scaffolding on which the decomposition or projection view of ranking is built.

In the exterior calculus view of partial differential equations, the main objects are often differential forms. In finite element and discrete exterior calculus these are usually discretized as cochains on simplicial complexes. The boundary and coboundary operators are used in defining differential operators, and are the building blocks of higher order operators like Laplacians. A similar situation holds for graphs treated as abstract simplicial complexes. One difference from partial differential equations is the absence of any geometric, i.e. metric, information. The vertices of the graphs in the ranking problem need not be placed in any particular geometric location. The metric information in the differential equations case is captured in the Hodge star operator, which we will not have occasion to use in the ranking problem on graphs.

The *boundary* operator $\partial_p : C_p(G) \to C_{p-1}(G)$ is usually described by first defining it on p-simplices and then extending it to $C_p(G)$. In the elementary chain basis it takes the form of a matrix with entries that are either 0 or ± 1. For our purpose, we take the simpler route and define these directly as matrices. The first of these is simply the vertex-edge adjacency matrix of graph theory. This has one column for each edge, with a -1 for the starting node and 1 for the ending node of that edge. This is the matrix form of ∂_1 in the elementary chain basis. Similarly there is an edge-triangle adjacency matrix ∂_2. Each column in it corresponds to a triangle and there is a ± 1 for each of the three edges that appear in the triangle. The entry is a $+1$ if the triangle and edge orientations match, and a -1 if the orientations do not match. Note that in a general graph an edge may appear in any number of triangles. This is different from the simplicial approximation of manifolds, i.e. meshes, that are used in partial differential equations. This will play an important role in the performance of linear solvers which were originally designed for solving partial differential equations on meshes.

The vector spaces and the boundary maps are arranged in what is known as a *chain complex*

$$0 \longrightarrow C_2(G) \xrightarrow{\partial_2} C_1(G) \xrightarrow{\partial_1} C_0(G) \longrightarrow 0 \tag{1}$$

where the first and last maps are zero operators. The most important fact about the boundary operators is that $\partial_p \circ \partial_{p+1} = 0$. It is the crucial fact needed in the decomposition described in Section 5. Analogous to the chain complex is the *cochain complex* which is arranged in the reverse order and uses the *coboundary* operator. Since we are dealing with real-valued chains and cochains, the coboundary operator δ_p is simply the dual of the boundary operator ∂_{p+1} and is defined by requiring

$$(\delta_p \alpha)(c) = \alpha(\partial_{p+1} c),$$

for all p-cochains α and $(p+1)$-chains c. This is more suggestive when the evaluation of a cochain on a chain is written as a pairing. Then the above relation can be written as $\langle \delta_p \alpha, c \rangle = \langle \alpha, \partial_{p+1} c \rangle$. When the elementary cochain basis is used, the matrix form of δ_p is simply ∂_{p+1}^T so we will use the transposed boundary matrix notation rather than the δ notation. We will write the cochain complex as

$$0 \longrightarrow C^0(G) \xrightarrow{\partial_1^T} C^1(G) \xrightarrow{\partial_2^T} C^2(G) \longrightarrow 0 \tag{2}$$

in which clearly $\partial_2^T \circ \partial_1^T = 0$ because of the analogous property of the boundary matrices.

4.4 Homology and cohomology

A fundamental problem in topology is to determine if two given spaces are topologically the same (homeomorphic) or different. We will recall the definitions of homology and cohomology, which are convenient tools for distinguishing spaces. Two spaces whose homology or cohomology differs are not homeomorphic. We need these notions in order to discuss our experiments on the topology of clique complexes in Section 7. As in the previous section, we will use real-valued chains and cochains.

The space of *p-cycles* is the space $\ker\partial_p$ (kernel of ∂_p), which is a subspace (as a vector space) of $C_p(G)$. The image of the boundary map coming from the $p+1$ dimension in a diagram like (1) is $\operatorname{im}\partial_{p+1}$ (image of ∂_{p+1}), which is also a subspace of $C_p(G)$. Their quotient $\ker\partial_p / \operatorname{im}\partial_{p+1}$, in the sense of vector spaces is called the p-dimensional *homology* space and denoted $H_p(G)$. Thus elements of $H_p(G)$ are equivalence classes of cycles. Cycles b and c are in the same class if $b-c$ is in $\operatorname{im}\partial_{p+1}$, and then b and c are said to be *homologous* to each other.

Remark 4.2. If the values of the chains (which are called coefficients) need to be emphasized, one writes $H_p(G;\mathbb{R})$ or $H_p(G;\mathbb{Z})$ for real or integer homology, respectively, and so on. Integer homology captures more information than real homology [48]. However, the real homology does include the one piece of information that is useful for interpreting our experiments in Section 7. The number called *Betti* number β_p for p-dimension, which is usually defined in integer homology, turns out to be the same as the vector space dimension of $H_p(G;\mathbb{R})$. This is a consequence of the universal coefficient theorem of algebraic topology [48, Chapter 7], or more simply from [48, Theorem 11.4].

On the cochain side one has the corresponding p-dimensional *cohomology* space $H^p(G)$, defined as the quotient space $\ker\partial_{p+1}^T / \operatorname{im}\partial_p^T$. As vector spaces, $H^p(G)$ and $H_p(G)$ are isomorphic. This is because their dimensions are the same, which follows easily from the rank-nullity theorem of linear algebra, and the basic facts about the four fundamental subspaces.

4.5 Laplace-deRham operators

The cochain and chain complexes can be combined, and excursions in that diagram lead to various Laplacian operators. The combined diagram that will suffice for this paper is

$$\begin{array}{ccccc} C^0(G) & \xrightarrow{\partial_1^T} & C^1(G) & \xrightarrow{\partial_2^T} & C^2(G) \\ \updownarrow & & \updownarrow & & \updownarrow \\ C_0(G) & \xleftarrow{\partial_1} & C_1(G) & \xleftarrow{\partial_2} & C_2(G) \end{array} \tag{3}$$

where the vertical arrows are vector space duality isomorphisms. The excursions used to define the new operators start at one of the cochains and traverse the box or boxes on the left, right, or both sides. In differential geometry and Hodge theory the resulting operators are known as the *Laplace-deRham* operators [1] and denoted Δ_p if they act on differential p-forms. In numerical analysis, these operators are increasingly being referred to as the *Hodge Laplacians* [3, 4]. The corresponding diagram in differential geometry consists of differential forms at both levels. The vertical arrows in that case are the Hodge star operators which contain the metric information about the manifold.

In the present case, by identifying $C^p(G)$ and $C_p(G)$ via vector space duality isomorphisms, we can abuse notation and use the identity operator for the vertical arrows. Three different Laplace-deRham operators $\Delta_p : C^p(G) \to C^p(G)$ can be defined for graphs augmented with triangles and all are of interest. These are

$$\Delta_0 = \partial_1\,\partial_1^T \qquad \Delta_1 = \partial_1^T\,\partial_1 + \partial_2\,\partial_2^T \qquad \Delta_2 = \partial_2^T\,\partial_2\,. \tag{4}$$

If cliques with more than 3 vertices were also to be included, then the definition of Δ_2 would change to $\partial_2^T\partial_2 + \partial_3\partial_3^T$ and there would be a Δ_3, Δ_4 and so on. From the definitions of these operators it is clear that the matrix form of any Laplace-deRham operator is square and symmetric.

Remark 4.3. When we study topology of clique complexes, the main objective will be to measure the 1-dimensional integer homology Betti number β_1. Using basic linear algebra combined with Hodge decomposition of p-cochains we give here a simple proof that $\dim\ker\Delta_p = \beta_p$. We will show that

$$\dim\ker\Delta_p = \dim H_p(G;\mathbb{R})\,,$$

as vector spaces. Then by Remark 4.2 the desired result follows. Recall that we use N_p for the number of p-simplices in the clique complex of G, and this number is the same as $\dim C_p = \dim C^p$. By Hodge decomposition of p-cochains (see Section 5) we have

$$\begin{aligned}\dim\ker\Delta_p &= \dim C^p - \dim\operatorname{im}\partial_p^T - \dim\operatorname{im}\partial_{p+1}\\ &= N_p - \dim\ker\partial_p^\perp - \dim\operatorname{im}\partial_{p+1}\\ &= N_p - (N_p - \dim\ker\partial_p) - \dim\operatorname{im}\partial_{p+1}\\ &= \dim\ker\partial_p - \dim\operatorname{im}\partial_{p+1} = \dim H_p(G;\mathbb{R}) = \beta_p\,.\end{aligned}$$

4.6 Vector calculus analogies

The matrix ∂_1^T is a graph analog of the gradient and ∂_1 is the analog of negative divergence. Similarly ∂_2 is the two-dimensional vector curl and ∂_2^T is the two-dimensional scalar curl [28]. The two-dimensional vector calculus diagram analogous to (3) is

$$\text{functions} \underset{-\operatorname{div}}{\overset{\operatorname{grad}}{\rightleftarrows}} \text{vector fields} \underset{\mathbf{curl}}{\overset{\operatorname{curl}}{\rightleftarrows}} \text{densities} \tag{5}$$

since divergence is the negative adjoint of gradient, and scalar and vector curls are adjoints of each other in two dimensions. This is an example of a de Rham complex [4, 13]. The more familiar three-dimensional vector calculus has the de Rham complex given below.

$$\text{functions} \underset{-\operatorname{div}}{\overset{\operatorname{grad}}{\rightleftarrows}} \text{vector fields} \underset{\operatorname{curl}}{\overset{\operatorname{curl}}{\rightleftarrows}} \text{vector fields} \underset{-\operatorname{grad}}{\overset{\operatorname{div}}{\rightleftarrows}} \text{densities} \tag{6}$$

Just as $\partial_2^T\,\partial_1^T = 0$ in the cochain complex, $\operatorname{curl}\circ\operatorname{grad} = 0$ and $\operatorname{div}\circ\operatorname{curl} = 0$ in these diagrams above.

The 0-Laplacian in (4) is the discrete analog of the usual scalar function Laplacian in vector calculus and is also the combinatorial graph Laplacian (without normalization, see [21]). For the de Rham complex (5) $\Delta_0 = -\operatorname{div}\circ\operatorname{grad}$. The 1-Laplacian in (4) is the discrete analog of the vector Laplacian $\Delta_1 = \mathbf{curl}\circ\operatorname{curl} - \operatorname{grad}\circ\operatorname{div}$ in the de Rham complex (5). If we did not include the triangles (or cells) and considered G only as a 1-dimensional complex, ∂_2 would be the zero matrix. Then the 1-Laplacian would be $\partial_1^T\,\partial_1$ which is sometimes called the edge Laplacian in graph theory. There is no name for Δ_2 in graph theory. But this 2-Laplacian is the graph theoretic analog of the 2-Laplacian in Hodge theory [1] and finite element exterior calculus [4] on a 2-dimensional manifold.

5 Hodge Decomposition of Vector Spaces

Hodge decomposition is an important tool in computer graphics [60], engineering [19] and mathematics [1, 47]. It generalizes the well-known Helmholtz decomposition of vector fields in Euclidean space [19] to differential forms on manifolds. The Helmholtz decomposition states that every vector field on a compact simply connected domain can be decomposed into a gradient of a scalar potential and a curl of a vector potential. The decomposition is orthogonal and hence unique although the potentials are not. The first part is curl-free and the second part is divergence-free. If the domain has nontrivial 1-dimensional homology (e.g., if it is an annulus, or a torus) then a third component called the harmonic vector field arises.

For finite-dimensional vector spaces, Hodge decomposition is a really simple idea. It is just the "four fundamental spaces" idea (popularized in Professor Strang's books) taken one step further. For a matrix A with m rows and n columns, the four fundamental subspaces are the column space of A, the nullspace

of A, the row space of A (which is the column space of A^T) and the left nullspace of A (which is the nullspace of A^T) [57, page 90]. We slightly prefer the terminology used in algebra where these would be denoted $\operatorname{im} A$, $\ker A$, $\operatorname{im} A^T$, and $\ker A^T$ and referred to as the image of A, kernel of A, image of A^T, and kernel of A^T.

Let U, V and W be finite-dimensional inner product vector spaces. Let $A : U \to V$ and $B : V \to W$ be linear maps such that $B \circ A = 0$. Define $\Delta := AA^T + B^T B$. The vectors in $\ker \Delta$ are called *harmonic*. Pictorially, we have

$$U \underset{A^T}{\overset{A}{\rightleftarrows}} V \underset{B^T}{\overset{B}{\rightleftarrows}} W \tag{7}$$

More formally, the transposes are the adjoint operators which are maps between the vector space duals, and adjointness requires the presence of inner products. For example, $A^T : V^* \to U^*$, where V^* and U^* are the vector space duals of the corresponding spaces. However, our inner products will always be the standard dot product, and we will identify the vector spaces and their duals. Thus we can get away with the slightly informal notation used in the diagram above.

Whenever we have a situation as in (7) above, the middle space splits into three subspaces. A splitting of V into two parts is just a consequence of the fact that V consists of the subspace $\operatorname{im} A$ and its orthogonal complement $\operatorname{im} A^\perp = \ker A^T$. The presence of the second map B and the fact that $B \circ A = 0$ is the crucial ingredient for getting a further splitting of $\ker A^T$. Just as the first split comes from two of the fundamental subspaces, the finer splitting of one of the pieces is yet another use of the fundamental subspaces ideas. These ideas are made more precise in the following elementary fact.

Fact 5.1. There exists a unique orthogonal decomposition of V (called the Hodge decomposition) as:

$$V = \operatorname{im} A \oplus \operatorname{im} B^T \oplus \ker \Delta \,.$$

Moreover, $\ker \Delta = \ker B \cap \ker A^T$.

Proof. We have first the obvious decomposition $V = \operatorname{im} A \oplus (\operatorname{im} A)^\perp$, where $(\operatorname{im} A)^\perp$ means the orthogonal complement of $\operatorname{im} A$. Thus $V = \operatorname{im} A \oplus \ker A^T$, from which follows that $V = \operatorname{im} A \oplus \operatorname{im} B^T \oplus ((\operatorname{im} B^T)^\perp \cap \ker A^T)$. This is due to the fact that $A^T \circ B^T = 0$ because of which $\operatorname{im} B^T \subset \ker A^T$. This finally yields $V = \operatorname{im} A \oplus \operatorname{im} B^T \oplus (\ker B \cap \ker A^T)$. To prove that $\ker \Delta = \ker B \cap \ker A^T$, it is trivial to verify that $\ker B \cap \ker A^T \subset \ker \Delta$. For the other direction, let $h \in \ker \Delta$. Then $0 = \langle \Delta h, h \rangle = \langle A^T h, A^T h \rangle + \langle Bh, Bh \rangle$ from which the result follows. Here the three inner products above are on V, U and W, respectively. □

To be precise, one should write $V \cong \operatorname{im} A \oplus \operatorname{im} B^T \oplus \ker \Delta$, since $B^T : W^* \to V^*$. However, we will continue to use equality by identifying the dual spaces V^* with the corresponding original vector spaces V etc. as mentioned earlier.

6 Least squares ranking and Hodge decomposition

We first consider the case when a given pairwise data, i.e., cochain $\omega \in C^1(G)$ has components along both $\operatorname{im} \partial_1^T$ and $\operatorname{im} \partial_2$. In this case, the Hodge decomposition, least squares, normal equations, and the Karush-Kuhn-Tucker equations are equivalent. This is the content of Theorem 6.2. The restrictions can be dropped to prove analogous theorems involving fewer equations.

In a least squares problem $Ax \simeq b$, to minimize $\|b - Ax\|_2^2$ as a function of x, a necessary condition is that the gradient be zero which yields the normal equations. Thus residual minimization implies the normal equations. For the converse, often a sufficient condition that is described in text books is that the Hessian matrix (which is $2A^T A$, in this case) be positive definite (see for example, [33, page 110]). This is often useful in the classical least square case in which $m \geq n$. For then, if A is full rank $A^T A$ is

positive definite. In our case, the matrices $A^T A$ will be $\partial_1\,\partial_1^T$ or $\partial_2^T\,\partial_2$. In most complexes with interesting topology we *cannot* rely on these to be nonsingular since ∂_1^T and ∂_2 will have nontrivial kernels. The constant functions on the vertices constitute $\ker\partial_1^T$ and all spheres are in $\ker\partial_2$. As an alternative, we will use the following lemma.

Lemma 6.1. *Given a matrix $A \in \mathbb{R}^{m\times n}$ and a vector $x_* \in \mathbb{R}^n$, if $A^T A x_* = A^T b$ and $x_* \notin \ker A$ then x_* minimizes the residual norm $\|b - Ax\|_2$ over all $x \in \mathbb{R}^n$.*

Proof. The dot product $\langle b - Ax_*, Ax_*\rangle = \langle A^T(b - Ax_*), x_*\rangle = 0$. This means either $b - Ax_* = 0$, in which case we are done, or the vectors $b - Ax_*$ and Ax_* are orthogonal. The latter means that the shortest distance from b to $\operatorname{im} A$ is achieved by Ax_*. □

Theorem 6.2 (Basic Fact)**.** *Given ω, $h \in C^1(G)$, $\alpha \in C^0(G)$, $\beta \in C^2(G)$, with $\alpha \notin \ker\partial_1^T$ and $\beta \notin \ker\partial_2$, the following are equivalent.*

(i) Hodge Decomposition (HD)

$$\begin{aligned} &\omega = \partial_1^T\,\alpha + \partial_2\,\beta + h\,, \\ &h \in \ker\Delta_1\,. \end{aligned} \tag{8}$$

(ii) Least Squares (LS)

$a = \alpha, b = \beta$, and $s = h$ are optimal values of the two least squares problems

$$\min_a \|r\|_2 \quad \textit{such that} \quad r = \omega - \partial_1^T\,a\,, \tag{9}$$

$$\min_b \|s\|_2 \quad \textit{such that} \quad s = r_* - \partial_2\,b\,, \tag{10}$$

where r_* is the minimizing residual for (9)*. In least squares short hand notation one would write the two problems as $\partial_1^T\,a \simeq \omega$ and $\partial_2\,b \simeq r_*$.*

(iii) Normal Equations (NE)

$a = \alpha$ and $b = \beta$ are a solution of the two linear systems

$$\partial_1\,\partial_1^T\,a = \partial_1\,\omega\,, \tag{11}$$

$$\partial_2^T\,\partial_2\,b = \partial_2^T\,r_*\,, \tag{12}$$

where r_* is the residual $\omega - \partial_1^T\,\alpha$.

(iv) Karush-Kuhn-Tucker Equations (KKT)

$a = \alpha, b = \beta$, and $s = h$ are a solution of the two saddle-type systems

$$\begin{bmatrix} I & \partial_1^T \\ \partial_2^T & 0 \end{bmatrix} \begin{bmatrix} r \\ a \end{bmatrix} = \begin{bmatrix} \omega \\ 0 \end{bmatrix}, \tag{13}$$

$$\begin{bmatrix} I & \partial_2 \\ \partial_1 & 0 \end{bmatrix} \begin{bmatrix} s \\ b \end{bmatrix} = \begin{bmatrix} r_* \\ 0 \end{bmatrix}, \tag{14}$$

where r_* is part of the solution for the first system.

Proof. Follows from Lemma 6.1, elementary calculus and linear algebra. □

Remark 6.3. One can prove analogous theorems for the case when $\alpha \in \ker \partial_1^T$ and/or $\beta \in \ker \partial_2$. This would involve skipping the equations corresponding to the term that is in the kernel. If both are, then the given data is purely harmonic.

Remark 6.4. The existence of the Hodge decomposition comes from Fact 5.1. The theorem above states the equivalence of Hodge decomposition with least squares, normal and Karush-Kuhn-Tucker equations.

6.1 Implications of orthogonality

The three terms in the Hodge decomposition (8) are mutually orthogonal. This is easy to see. It follows simply from the fact that $\partial_1 \partial_2 = 0$ and from the definition of the harmonic part ($\ker \Delta_1$). For example, given an $\omega \in C^1(G)$, if it has a nonzero harmonic part h, then $\langle h, \partial_1^T \alpha \rangle = \langle \partial_1 h, \alpha \rangle = 0$ since h is in $\ker \partial_1 \cap \ker \partial_2^T$.

Due to these orthogonality conditions, it is easy to see that the second least squares problem, which is $\partial_2 b \simeq r_*$, can also be written as $\partial_2 b \simeq \omega$. Similar changes can be made from r_* to ω in the second systems in all the formulations above. For ease of reference, below we write the least squares and normal equations using ω instead of r_* all in one place. The least squares systems are

$$\partial_1^T a \simeq \omega, \tag{15}$$

$$\partial_2 b \simeq \omega, \tag{16}$$

and the corresponding normal equations

$$\partial_1 \partial_1^T a = \partial_1 \omega, \tag{17}$$

$$\partial_2^T \partial_2 b = \partial_2^T \omega. \tag{18}$$

Note from the definition of the Hodge Laplacians in (4) that the above normal equations can be written as $\Delta_0 a = \partial_1 \omega$ and $\Delta_2 b = \partial_2^T \omega$.

6.2 Connection with optimal homologous chains problem

Notice that the least squares formulations in (9) and (10) are exactly analogous to the optimal homologous chain problem from [23]. For example, in (9) the given cochain is ω and one is looking for the smallest cochain r which is cohomologous to it. In [23] it is the 1-norm of chains that is minimized over all homologous chains. In contrast, here the 2-norm of *co*chains is minimized over all *co*homologous cochains. This results in our solving linear systems in this paper as opposed to linear programming which is used in [23]. In spite of these differences between the two problems, it is interesting that the problem of least squares ranking on graphs and a fundamental problem of computational topology (computing optimal homologous chains) are related.

6.3 Interpretation in terms of ranking

Given any pairwise comparison data $\omega \in C^1(G)$ we see from the Theorem 6.2 that there exists a cochain $\alpha \in C^0(G)$ (the vertex potential or ranking), a cochain $\beta \in C^2(G)$, and a harmonic field $h \in \ker \Delta_1$, such that $\omega = \partial_1^T \alpha + \partial_2 \beta + h$. The α term is the scalar potential that gives the ranking. The β term is defined on cells and captures the local inconsistency in the data. The harmonic part contains the inconsistency that is present due to loops longer than the maximum number of sides in the cells. If only 3-cliques (triangles) are considered as the 2-dimensional cells, then any inconsistency in loops of length four or more will be captured in the harmonic part. The following example should make some of this more apparent.

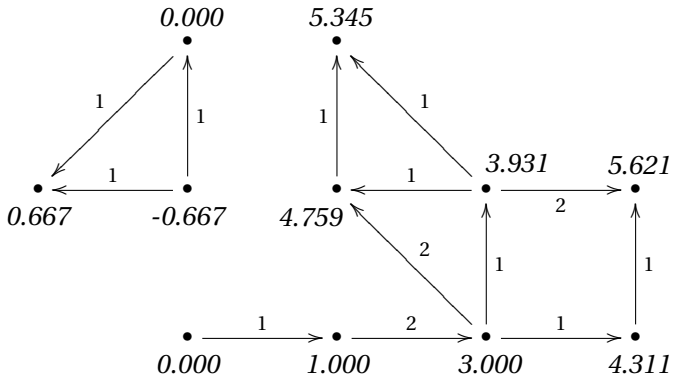


Figure 1: The results of the first least squares problem (9) on a graph.

Example 6.5. Figure 1 shows an example of solving the first least squares problem (9). It is just as easy to work with disconnected graphs, so we show a graph with two components. The values on the edges is the given data ω in $C^1(G)$. The vertex potential values $\alpha \in C^0(G)$ are written in italics. Note that in the straight line part of the graph which does not involve a cycle, it is clear what the vertex potential should be (up to an additive constant). There will be no residual in this case. The first triangle after the straight line part is consistent because the value on the hypotenuse is the sum of the values on the other two sides which are oriented appropriately. The other triangles and the square loop are all inconsistent. Here only triangles are chosen as the 2-dimensional cells, so the β part will be the inconsistency associated with the triangles if the second problem were also to be solved. The harmonic part h would be the inconsistency in the square loop. Note that because of two connected components the dimension of $\ker \Delta_0 = \partial_1 \partial_1^T$ will be two. Fixing one vertex value in each of the two components and deleting the appropriate row and column will make the normal equations system (11) nonsingular. We will discuss the issue of nontrivial kernels in the context of linear solvers in Section 8.

7 Experiments on Topology of Clique Complexes

A recent paper by Kahle [37] explores the homology of clique complexes arising from Erdős-Rényi random graphs. The number of connected components of a graph is the same as the dimension of its 0-dimensional homology. The connectedness of random graphs has been explored in literature for many years and the study of higher dimensional homology can be considered as the new and natural extension of that line of research. For the ranking problem, 1-homology is of particular interest. When 1-homology of a graph is trivial, then by Remark 4.3 there cannot be any harmonic component in any 1-cochain. Then if there are no local inconsistencies (i.e., the curl part is zero) a 1-cochain will be a pure gradient, hence globally consistent.

Kahle provides bounds on the edge density for Erdős-Rényi graphs for which the p-dimensional homology is almost always trivial, and bounds for which it is almost always nontrivial. Since all results of this type are "almost always", in the rest of this section, we will omit that phrase. We restrict our attention to 1-homology, and the 2-dimensional clique complex of the graph is the relevant object. In this setting, Kahle states that for an Erdős-Rényi graph with n nodes and edge density ρ, the 1-homology will be trivial when $\rho < 1/n$ or when $\rho > 1/\sqrt[3]{n}$. The 1-homology will be nontrivial when $1/n < \rho < 1/\sqrt{n}$. When $1/\sqrt{n} < \rho < 1/\sqrt[3]{n}$ there is a theoretically undetermined transition in homology. (There is a typo in Jiang et al.'s quotation of the bounds from Kahle's results.)

The numerical framework for exploring least squares ranking on graphs can be applied to study

clique complexes, almost without any changes. For example, it is satisfying to see Kahle's bounds appear in the experimental results shown in the first column of Figure 3. But what is more interesting is that experiments like these can serve as tools for developing new conjectures and asking more detailed questions than are answered by the current theory. Once a conjecture looks numerically plausible, one can set about trying to find a mathematical proof. But before we mention some such questions, we note that the apparent violation of Kahle's bounds in Figure 3 is not really a violation, because his bound are true in the limit as the number of vertices n goes to infinity. For example, there are some nonzero homology points in the region that is supposed to almost always have trivial homology.

Using this experimental tool one can ask new questions, such as, what is the behavior of the 1-dimensional Betti number as a function of ρ? By Remark 4.3, the Betti number can be measured by measuring the dimension of the space of harmonic cochains, i.e., $\dim\ker\Delta_1$. Since $\ker\Delta_1 = \ker\partial_1 \cap \ker\partial_2^T$, one can measure the Betti number by stacking the matrices for ∂_1 and ∂_2^T, one on top of the other and the kernel dimension of this matrix yields the desired Betti number. The kernel dimension can be found by computing a singular value decomposition and counting the number of zero singular values. A faster alternative is to compute the number of zero eigenvalues of Δ_1 by using a sparse eigensolver. Usually, the distinction between what should be considered nonzero and what should be considered zero is very evident in our experiments.

Results about the Betti number give a more nuanced picture than the presence or absence of homology. For example, in terms of Betti number, one can ask if the transition from nonzero homology region to the zero homology region on the right is sudden. Our experiments on Erdős-Rényi graphs shed some light on these questions. In Figure 3, the bottom graph in the first column shows how the Betti number varies as a function of ρ. A clear trend is visible and the Betti number appears to peak at the start of the transition zone where the theory is silent (the bounds of Kahle are marked as dashed vertical lines). The quantification of homology can also take another form in these experiments. One can investigate how much of the norm of a 1-cochain is contained in the harmonic component, when the homology is nontrivial. Or, how does this harmonic component vary as a function of ρ? The results are in the top two graphs in the left column of Figure 3. The right column shows results of experiments on topology of clique complex of Barabási-Albert scale-free graphs, for which no theory has yet been developed. Although Kahle's bounds were not developed for Barabási-Albert graphs, we have drawn those for the plots corresponding to Barabási-Albert graphs in order to provide context and comparison with the Erdős-Rényi results.

8 Comparing Linear Solvers

In most areas of numerical analysis, the computation that sits at the heart of the solution method is usually the solution of a linear system. Least squares ranking on graphs is no different. This section is about our numerical experiments for testing the accuracy and speed of linear system solvers for the first and second least squares problems of ranking on graphs.

A small part of our experimental work was on graphs with special structure, such as path, cycle, star, and wheel graphs. However, our main focus has been on two popular random graph models, and on scale-free graphs. The earliest and perhaps most studied random graph model is the one of Erdős and Rényi [11, 25]. The two parameters for this model are the number of nodes and the probability that any pair of nodes is connected by an edge. We will refer to this probability as the *edge density*. This model does not account for the phenomena of clustering that is seen in many networks in societies and this shortcoming is overcome by the model of Watts and Strogatz model [49]. However, neither of these account for the power-law shape of degree distributions that are seen in many real-life graphs such as the World Wide Web, reaction networks of molecules in a cell, airline flight networks and so on. These are

called scale-free networks or graphs and this feature is captured by the model of Barabási and Albert [5]. The generative model is often implemented as a random process, although in practice these are typically not random graphs.

In the numerical analysis community there is a lot of accumulated experience on solving linear systems that arise from partial differential equations. Studies of systems arising from graphs are less common but appearing with increasing frequency. For least squares ranking on general graphs there is no guidance available in the literature. Ours is a first step in an attempt to fill that gap. The fact that the underlying problem is coming from a graph introduces some new challenges as we will demonstrate via our experiments in this section.

We only consider iterative linear solvers here, though sparse direct solvers might be worth considering. We used a variety of iterative Krylov methods suitable for symmetric systems and one that is suitable for rectangular systems. We also used algebraic multigrid using smoothed aggregation and Lloyd aggregation. The results are discussed in Sections 8.3–8.5. An especially attractive feature of all these methods is their ability to ignore the kernel of the operator involved. We do not know how the direct methods could be made to do that for the second least squares problem which can have a large dimensional kernel depending on the graph topology. In direct solvers, for the first least squares problem the nontrivial kernel can be handled by fixing the value at a single vertex, just as is done by fixing pressure at a point in fluid problems.

Remark 8.1. For an arbitrary graph G, by Theorem 6.2 (or its special cases mentioned in Remark 6.3), the first least squares problem of ranking (15) is equivalent to the the normal equation (17). But the matrix involved is then $\partial_1 \partial_1^T$ which is the combinatorial graph Laplacian Δ_0 and hence it is symmetric and diagonally dominant. Thus, it can be solved by the method of Koutis et al. [40], which can solve such systems in time approaching optimality. However, at the time of writing, no reliable implementation of the method of Koutis et al. was available. The system matrix Δ_2 for a general graph need not be diagonally dominant. Consider for example the complete graph K_5 in which every 3-clique is taken to be a triangle. It is easy to verify that Δ_2 is not diagonally dominant. It is a 10×10 matrix with 3s along the diagonal and with each row containing six off-diagonal entries that are ± 1 (with four entries that are 1 and two that are -1). Thus, for a general graph, the Koutis et al. solver cannot be used for solving the second least squares problem of ranking due to lack of diagonal dominance.

8.1 Methodology

All numerical experiments were done using the Python programming language. The Krylov linear solvers used were those provided in the SciPy module [36]. The algebraic multigrid used was the one provided in PyAMG [9]. The simplicial complexes and boundary matrices were created using the PyDEC module [7]. The errors and times required by various solvers is generated as an average over multiple trials. This is done to minimize the influence of transient factors that can affect performance of a computer program. All experiments were carried out on an Apple MacBook with a 2.53 GHz Intel Core 2 Duo processor and with 4 GB of memory.

In each case, a graph G with the desired number of nodes N_0 and other desired characteristics (such as edge density in the case of Erdős-Rényi graphs) is first generated by a random process. We then find all the 3-cliques in the graph and create a simplicial complex data structure for the resulting 2-complex. Let the number of edges and triangles in G be N_1 and N_2.

A random ranking problem instance is created for this complex. This entails creating a random 1-cochain representing the comparison data on edges. The point of these experiments is to compare the accuracy and efficiency of the tested methods, and so the Hodge decomposition of this 1-cochain has to be known in advance. In other words, a random problem instance is a 1-cochain ω such that there are random but known $\alpha \in C^0(G)$, $\beta \in C^2(G)$, and $h \in \ker \Delta_1$ with $\omega = \partial_1^T \alpha + \partial_2 \beta + h$.

It is clear how to create the random gradient part $\partial_1^T \alpha$ and the random curl part $\partial_2 \beta$ – simply pick a random vector with N_0 entries for α and a random vector with N_2 entries for β. To compute a random harmonic part, one can compute the Hodge decomposition of a random 1-cochain ρ by solving two least squares problems $\partial_1^T a \simeq \rho$ and $\partial_2 b \simeq \rho$ as outlined in Section 6.1. If α and β are the solutions, then $\rho - \partial_1^T \alpha - \partial_2 \beta$ is a desired random harmonic cochain.

For variety we show here another method, and this is the one we used. It relies on the basic fact that the the residual $b - Ax$ in a least squares problem $Ax \simeq b$ is orthogonal to the $\operatorname{im} A$ and hence in $\ker A^T$. We pick a random 1-cochain ρ, that is, a random vector with N_1 entries. We then solve the single least squares problem

$$\begin{bmatrix} \partial_1^T & \partial_2 \end{bmatrix} x \simeq \rho \,, \tag{19}$$

where the matrix $[\partial_1^T \ \partial_2]$ is formed by horizontally stacking ∂_1^T and ∂_2 matrices. If x is the solution of this least squares problem, then the residual $\rho - [\partial_1^T \ \partial_2]\, x$ is harmonic. This is because

$$\rho - [\partial_1^T \ \partial_2]\, x \in \ker [\partial_1^T \ \partial_2]^T = \ker \begin{bmatrix} \partial_1 \\ \partial_2^T \end{bmatrix} = \ker \partial_1 \cap \ker \partial_2^T = \ker \Delta_1 \,.$$

8.2 Spectral analysis

Recall that if A is a symmetric positive definite matrix, the number of conjugate gradient iterations is related to the norm of the error by the inequality

$$\frac{\|e_k\|_A}{\|e_0\|_A} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k . \tag{20}$$

See for example [32, page 51]. Here $\|x\|_A$ is the A-norm of x, i.e., $\|x\|_A^2 := \langle x, Ax \rangle$ and $\kappa = \lambda_{\max} / \lambda_{\min}$ is the condition number of A. Here $\lambda_{\max}$ and $\lambda_{\min}$ are the largest and smallest magnitude eigenvalues of A, respectively. The same result holds even if A is singular, as long as it is semidefinite, $\|x\|_A$ is considered a seminorm, and $\lambda_{\min}$ is defined to be the smallest (in magnitude) nonzero eigenvalue of A. Given a desired error ϵ (in the A-norm), one can find the number of iterations of conjugate gradient method required to achieve that error, by substituting ϵ for $\|e_k\|_A$ and solving for the smallest k which satisfies inequality (20). We will refer to this number as *conjugate gradient iterations* required to achieve error ϵ. Thus for an arbitrary graph G, the conjugate gradient iterations required for the first least squares problem of ranking is given by inequality (20) using $\lambda_{\min}$ and $\lambda_{\max}$ for the graph Laplacian Δ_0.

Much is known about spectrum of the graph Laplacian for various types of graphs [26, 43, 45, 46], including random graphs and scale-free networks [20]. These results usually involve some graph property. For example, for various types of special graphs, $\lambda_{\min}$ is often bounded in terms of edge connectivity – the minimum number of edges to be removed to disconnect a graph. Thus one can make predictions like single iteration convergence of conjugate gradient method in the case of complete graphs, and this is borne out by our numerical experiments. For some special graphs, some well known lower bounds or formulas for $\lambda_{\min}$ are given below. An easy upper bound for $\lambda_{\max}$ of Δ_0 is twice the maximum degree. This follows from Gerschgorin's theorem [61]. In the table below, $\eta(G)$ is the edge connectivity, a path graph is an acyclic graph in which all but two vertices have degree two, and the two "end" vertices each have degree one. A cycle graph is a 2-regular graph on which there is only one cycle of size n, and a star graph is one in which all but one vertex have degree 1, and each is connected to a "center" vertex which has degree $n-1$. Figure 5 in Appendix B shows comparisons of iteration bounds and actual iteration counts that occur in numerical experiments.

Type of graph	$\lambda_{\min}$
General	$2\eta(G)\,(1-\cos(\pi/n))$
Complete	n
Path	$2\,(1-\cos(\pi/n))$
Cycle	$2\,(1-\cos(2\pi/n))$
Star	1

For Erdős-Rényi, Watts-Strogatz, and Barabási-Albert graphs, there are fewer results for the spectral radii but there exist results bounding these from which iteration bounds can be obtained. Results of some experiments using such graphs are shown in Figure 4.

Remark 8.2. When the simplicial (or cell) 2-complex representation of G can be embedded as a meshing of a compact surface with or without boundary, we *can* place bounds on the spectrum of Δ_2. We do this by generating the *dual graph* G^D of G, in which every triangle of G is a vertex in G^D; the dual vertices in G^D are connected by edges which are dual to those in G. One can then bound the spectrum of Δ_2 in both the boundary and boundaryless surface cases using Cauchy's interlacing theorem for eigenvalues [53]. However, while any graph may be embedded in a surface of sufficiently high genus, the embedding of the simplicial 2-complex is not possible for *general* graphs. As an example, consider the complete graph K_5 introduced above, for which $N_0 = 5$, $N_1 = 10$, and $N_2 = 10$. The Euler characteristic of this complex is $\chi = N_0 - N_1 + N_2 = 5$. Assume that K_5 could be embedded on a surface of genus g and with b disjoint portions of boundary. Then $\chi = 2 - 2g - b$ which reduces to $2g + b = -3$ which is not true for any nonnegative g and b leading to a contradiction.

8.3 Iterative Krylov methods

The Krylov solvers that we used in our experiments are conjugate gradient (CG) and minimal residual (MINRES) for normal equations and saddle formulation [56, 62], and LSQR [51, 52] for the least squares system solved using the given rectangular matrix without forming the square system. There are many other Krylov solvers we did not test. For example, we did not test GMRES since all of our square linear systems are symmetric. We did not test CGLS or CGNE since they are mathematically equivalent to LSQR, which tends to be more commonly used. As mentioned earlier, the Laplacian systems will in general be symmetric positive semidefinite. Krylov methods have the nice property that they work in spite of a nontrivial kernel. This feature is especially effective in the graph problem because the matrices are integer matrices requiring no quadrature for their construction such as is required in the case of partial differential equations [10].

Tables 1–3 are timing and error results for various Krylov solvers on different formulations of the ranking problem. In each table, edge and triangle densities are with respect to the number of edges and all possible triangles in the complete graph. LSQR was used on the least squares equations directly, i.e., on the rectangular matrices in (15) and (16). The other solvers are used on the normal equations unless the '-K' designator is listed, which indicates the solver was used on the Karush-Kuhn-Tucker formulation (13) and (14). Reported errors are measured relative to the known exact solution except in cases identified by an asterisk ($*$) in $\|h\|$ column where the corresponding error is an absolute one. These are cases for which the homology of the simplicial 2-complex induced from the graph is trivial leading to a zero harmonic component. The relative error column reports error in the norm of the gradient part ($\|\partial_1^T \alpha\|$), norm of the curl part ($\|\partial_2 \beta\|$), and the norm of the harmonic part ($\|h\|$). The timing labeled α shows the iterations and time required for the first least squares problem, and the one labeled β shows these for the second least squares problem.

8.4 Algebraic multigrid methods

Multigrid methods work by creating a hierarchy of linear systems of decreasing sizes from the given problem. At any level in the hierarchy, a larger system is called a *fine grid* and a smaller system is called a *coarse grid*. The solutions of the different systems are related via prolongation (coarse to fine) and restriction (fine to coarse) maps. In geometric multigrid, coarser levels correspond to a coarser mesh. In *algebraic* multigrid, coarsening is carried out using only the matrix and an associated adjacency graph. Coarsening is performed by aggregating those vertices of this graph which have a strong connection. The strength of connection is defined in various ways for different schemes [58]. In smoothed aggregation, a vertex can belong fractionally to several aggregates [63]. In Lloyd aggregation, the number of connections between vertices and the centers of their aggregates is minimized [8].

We used algebraic multigrid with smoothed aggregation and Lloyd aggregation for the ranking problem on Erdős-Rényi and Watts-Strogatz random graphs, and Barabási-Albert scale-free graphs. The parameters for the various graphs were identical to ones used in the experiments with Krylov solvers. The results of these numerical experiments are given in Tables 4–6. The columns are the same as the ones in the Krylov tables 1–3. Only the normal equation formulations (17) and (18) are used in these experiments. In the Algorithm/Formulation column, the notation AMG (SA) and AMG (LA) are used to indicate the smoothed and Lloyd aggregation schemes. In this column, Schur indicates yet another variation which is described as Algorithm 1 in the next subsection. As before, in these tables, the cases where the homology of the simplicial 2-complex is trivial are marked by an asterisk ($*$). The cases where PyAMG failed in the setup phase are marked by a dagger symbol (†), and the cases where algebraic multigrid reached maximum number of specified iterations are marked by the double dagger symbol (‡). The entries marked with a dash symbol (–) are those for which Algorithm 1 cannot be applied because Δ_2 has no simple sparse/dense partitioning. Tables 7–9 list the measured setup and solve times for each of ranking problems on the three graph models where solution could be found using algebraic multigrid.

8.5 Schur complement and algebraic multigrid

A third approach is a hybrid one that combines algebraic multigrid with Krylov or direct solvers. The idea is to partition the problem into a small dense part and a large sparse part. The small part can be solved efficiently, e.g., by Krylov or direct methods, and the large sparse part can be solved by algebraic multigrid. A general matrix A is partitioned as

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where A_{11} is the large sparse part and A_{22} is the small dense part. The linear system $Ax = b$ can be written as

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

which can be reduced by row operations to

$$\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} - A_{21}A_{11}^{-1}A_{12} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 - A_{21}A_{11}^{-1}b_1 \end{bmatrix}. \tag{21}$$

One can then solve the second block for x_2 first and use the result to solve for x_1. This is the Schur complement formulation. The hope is that A_{22} will be small even if it is dense, and that A_{11} will be sparse. Then, if A_{11}^{-1} can be found easily, forming $A_{22} - A_{21}A_{11}^{-1}A_{12}$ will be easy and x_2 can be found perhaps using a dense but small linear system.

Algorithm 1 Iterative solution of Schur complement system using Krylov/direct and AMG solvers

Require: Approximate inverse $\widetilde{A}_{11}^{-1}$ and initial guess $x^{(0)}$.

1: Compute $r^{(0)} = \begin{bmatrix} r_1^{(0)} \\ r_2^{(0)} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} - \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \end{bmatrix}$

2: **for** $i = 0, 1, \ldots,$ until convergence **do**

3: Krylov/Direct-Solve: $(A_{22} - A_{21}\widetilde{A}_{11}^{-1}A_{12})e_2^{(i)} = r_2^{(i)} - A_{21}\widetilde{A}_{11}^{-1}r_1^{(i)}$

4: AMG-Solve: $A_{11}e_1^{(i)} = r_1^{(i)} - A_{12}e_2^{(i)}$

5: $x^{(i+1)} = x^{(i)} + e^{(i)}$

6: $r^{(i+1)} = b - Ax^{(i+1)}$

7: **end for**

In order to solve the dense block of (21), one needs the inverse of the sparse block A_{11} which can be computed using algebraic multigrid. Another approach is to approximate A_{11}^{-1} and to successively refine it by solving the two block equations of (21). A pseudocode description of this method is provided in Algorithm 1.

In general the matrix A will not already be in this sparse/dense partition form. Reordering can be performed using a simplistic technique such as sorting by degree of vertices, by reordering schemes like reverse Cuthill-McKee or approximate minimum degree ordering, or sorting by number of nonzeros in the rows. For a general graph Laplacian (corresponding to the normal equations of the first least squares system), sorting by degree of vertices is a more natural choice rather than reordering schemes that work well on meshes or graphs with uniform degree nodes. Likewise, for the second least squares problem, sorting by number of nonzeros in the rows is a natural choice for reordering. However, none of these schemes yield such a sparse/dense partitioning for the graphs we considered. See Figures 6–14 which show the patterns of nonzeros in Δ_0 and Δ_2.

Due to the nature of the nonzero patterns in the matrices and their reorderings, the computation of A_{11}^{-1} by algebraic multigrid is not feasible. This was borne out in our experiments and hence the details are not reported here. The results of using Algorithm 1 for the first least squares problem of ranking on Erdős-Rényi random graphs are shown in Table 4. While for this problem Algorithm 1 appears slightly more competitive than algebraic multigrid with smoothed aggregation or Lloyd aggregation, it is not at all competitive as compared to Krylov solvers. Thus, we did not apply it to the ranking problem on the other two graph models.

8.6 Discussion and comparisons

From the results in Tables 1–3, it is clear that amongst Krylov methods conjugate gradient is a good choice for the first least squares problem. For the second least squares problem, conjugate gradient performs better on sparser graphs but LSQR does better on denser graphs.

Algebraic multigrid performs optimally for certain types of elliptic partial differential equations on meshes. So a naive hope would be that it would do well with Laplacians on graphs. Our experiments demonstrate that although algebraic multigrid can sometimes solve the linear systems, it often performs quite poorly in terms of time as compared to Krylov methods. This is true even if the setup time required by algebraic multigrid is excluded. (The setup is the process of forming the coarser levels before solving can begin.) If the setup time is also included, the performance becomes much worse compared to Krylov methods. For example, in the cases corresponding to the last two rows of Table 4, algebraic multigrid could not even be used, whereas Krylov methods performed reasonably well (see Table 1). In the same table, for the β problem in rows 4 and 5, algebraic multigrid took between 2 and over 400 seconds while Krylov methods took only a fraction of a second. Even not counting the setup phase, these cases took

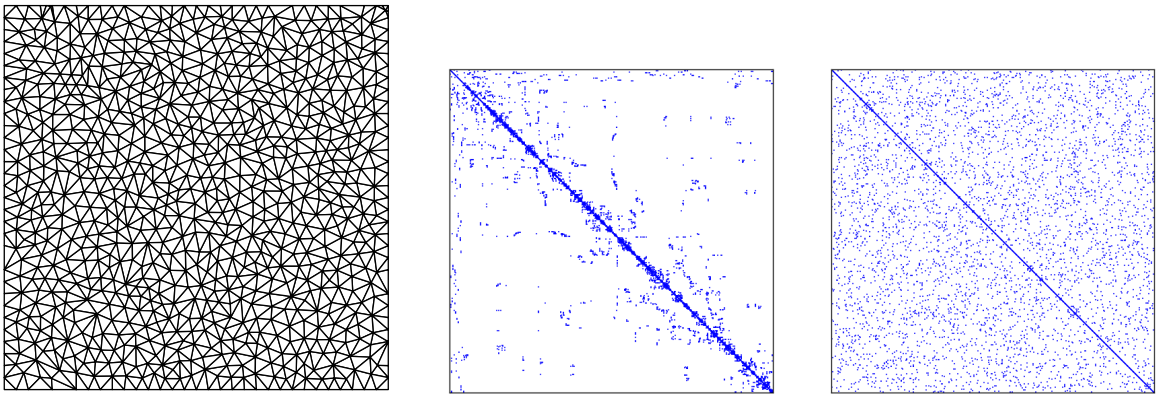


Figure 2: For the mesh shown on the left, the middle plot shows the nonzeros in Δ_0. The rightmost plot shows the nonzeros in Δ_0 for an Erdős-Rényi graph constructed using the same number of vertices and edges as are in the mesh. The Δ_0 on the mesh has locality whereas the one on the graph does not.

between over 1 to over 32 seconds for algebraic multigrid, as can be seen in Table 7. Similar behavior is seen for Barabási-Albert graphs. See for instance rows 4, 6, and 9 in Table 6.

We are making available the code, and providing details of the experiments, so that multigrid experts can help improve its performance. It is quite possible that the experts will find some small tweaks (which we missed, not being multigrid experts) that make algebraic multigrid competitive. In contrast, we note that the Krylov methods work well "out-of-the-box", for example without any preconditioning. One should keep in mind however, that our experiments are for small- to moderate-sized problems, those than can be done on a single serial computer. Currently that means a linear system with nearly a quarter million unknowns on a modest laptop. Moving to larger graph problems, requiring parallel computation, will open up a whole new field which we have not explored in this paper.

If the Schur complement method is used with the inverse matrix A_{11}^{-1} determined by algebraic multigrid then it is slower than using Krylov methods alone. This means that the time penalty in this method is attributable to either the matrix partitioning or to the algebraic multigrid solution for the inverse matrix, and not to the Krylov solution of the dense partition. The results of using Algorithm 1, which uses an approximate inverse, are shown in Table 4. It can be seen that for smaller graphs it is slower that just using algebraic multigrid and for larger graphs it is marginally faster. However, in either case it is much slower than using Krylov methods alone.

Tables 10–12 show the number of nonzero components in the linear systems being solved by the different methods. The matrix sizes can be determined based on the number of vertices, edges, and triangles. These tables show the details of all levels for multigrid methods. For the first least squares problem, the storage requirements for ∂_1 and Δ_0 are very similar, with only an additional N_0 nonzeros required in Δ_0 to store the vertex degrees. However, for the second least squares problem, the storage requirements for Δ_2 grow at a faster rate than for ∂_2. Figures 6–14 show a visual representation of the nonzeros in most of the Laplacian matrices. The original matrices as well as various reorderings are displayed in these figures. These figures indicate that aggregation schemes for algebraic multigrids may have to be rethought for these graph problems. There is no apparent (or very little) structure or locality visible in these figures. This is very different from what would be observed in the case of Laplacians on meshes. See for example Figure 2.

9 Conclusions and Future Work

Other researchers have formulated least squares ranking on graphs and studied its theory, applications, and connections with other fields. In this paper, we have presented the first detailed numerical studies of the two least squares problems involved. We also took this opportunity to introduce the problem in an elementary way and highlighted its many connections to different fields. For example, we have shown that in setting up the least squares ranking problem on graphs it is natural to make excursions into areas such as elementary exterior calculus and elementary algebraic topology. In fact, due to the absence of any geometry ranking provides an easy introduction to some basic aspects of exterior calculus and algebraic topology. The lack of geometry arises from the absence of location information about the vertices and the absence of a metric on the graphs. Since the problem and its formulation is so easy to state and work with, the first author has also used it in undergraduate projects with an aim to introducing undergraduate students to research. In contrast, the very closely related problems in elliptic partial differential equations are much harder to introduce at a comparable elementary level.

The same features which make it easy to state and formulate the problem also make it easy to implement the linear systems (i.e., construct the matrices involved) for these least squares problems. We have seen that the system matrices only involve the boundary matrices and their transposes, and these are easily constructed. Hodge decomposition and harmonic cochains on graphs are studied in this paper, but these can also be studied on a simplicial approximation of a manifold. There these problems arise in the context of numerical methods for elliptic partial differential equations. The presence of a metric in that case requires integration and the introduction of a matrix approximation of the Hodge star. This takes one into the realm of finite element exterior calculus or discrete exterior calculus. The creation of matrices (that are analogous to the ones studied in this paper) requires more work and functional analysis is required to fully appreciate the numerical issues like stability and convergence to smooth solution. In graph problems, there is no issue of convergence to a smooth problem because there is no such problem that we are approximating. The simple code developed for the ranking problem is also easy to use for experimental studies on topology of random clique complexes. This can be used to formulate conjectures, or as we have shown, to get more detailed information than the current theory in that field can provide.

Our detailed numerical studies lead to some interesting conclusions. Algebraic multigrid method is a topic of intense research currently since it has been very successful for certain elliptic partial differential equations. However, it performs poorly in the graph problems that arise in least squares ranking on graphs. At first this is surprising. After all, the two problems are Poisson's equations, one involving the scalar Laplacian and the other involving the Laplace-deRham operator on 2-cochains. However, as alluded to earlier and as can be seen from Figures 6–14, the linear systems in these graph problems suffer from lack of locality and structure which is usually apparent in partial differential equation problems on meshes.

For the first least squares problem, our results indicate that using the conjugate gradient solver on the normal equations is always the best option, generating the lowest errors and the fastest times for each of the three types of graphs we considered. For the second least squares problem, conjugate gradient is the better option when the triangle density (and thus density of Δ_2) is low, but as the triangle density increases, LSQR becomes the faster option.

An interesting area for future research is the decomposition of these problems into smaller pieces which can be solved with minimal interaction. One then also needs to combine the solutions thus obtained into a global solution. It is not at all obvious how this can be done. For partial differential equations, smaller subproblems are sometimes obtained by domain decomposition methods. Analogously, appropriate graph partitioning methods and their role in decomposing these graph problems may be worth studying. These might also be relevant in creating new aggregation schemes for algebraic multi-

grid. The connection between the solvers for diagonally dominant systems [40] and algebraic multigrid would also be interesting to study further. Once reliable software implementing the complete algorithm of [40] becomes available, comparisons between that and the various methods studied here would also be relevant. Graphs on surfaces have a special structure which can be exploited for efficiency. It would be interesting to study examples of ranking problems which are natural to study on surfaces. There are some hints that assignment of ranking work to committees might naturally fit into this framework.

Acknowledgement

The work of ANH and KK was supported in part by NSF Grant No. DMS-0645604. We thank Nathan Dunfield, Xiaoye Jiang, Rich Lehoucq, Luke Olson, Jacob Schroder, and Han Wang for useful discussions.

References

[1] ABRAHAM, R., MARSDEN, J. E., AND RATIU, T. *Manifolds, Tensor Analysis, and Applications*, second ed. Springer–Verlag, New York, 1988.

[2] AGARWAL, S. Learning to rank on graphs. *Machine Learning 81* (2010), 333–357. 10.1007/s10994-010-5185-8.

[3] ARNOLD, D. N., FALK, R. S., AND WINTHER, R. Finite element exterior calculus, homological techniques, and applications. In *Acta Numerica*, A. Iserles, Ed., vol. 15. Cambridge University Press, 2006, pp. 1–155.

[4] ARNOLD, D. N., FALK, R. S., AND WINTHER, R. Finite element exterior calculus: from Hodge theory to numerical stability. *Bull. Amer. Math. Soc. (N.S.) 47*, 2 (2010), 281–354. doi:10.1090/S0273-0979-10-01278-4.

[5] BARABÁSI, A., AND ALBERT, R. Emergence of scaling in random networks. *Science 286*, 5439 (1999), 509–512. doi:10.1126/science.286.5439.509.

[6] BARTHOLDI, III, J., TOVEY, C. A., AND TRICK, M. A. Voting schemes for which it can be difficult to tell who won the election. *Soc. Choice Welf. 6*, 2 (1989), 157–165. doi:10.1007/BF00303169.

[7] BELL, N., AND HIRANI, A. N. PyDEC: Algorithms and software for Discretization of Exterior Calculus, March 2011. Available as e-print on arxiv.org. arXiv:1103.3076.

[8] BELL, W. N. *Algebraic Multigrid for Discrete Differential Forms*. PhD thesis, University of Illinois at Urbana-Champaign, Urbana, Illinois, 2008.

[9] BELL, W. N., OLSON, L. N., AND SCHRODER, J. B. PyAMG: Algebraic multigrid solvers in Python v2.0, 2011. Release 2.0.

[10] BOCHEV, P., AND LEHOUCQ, R. B. On the finite element solution of the pure Neumann problem. *SIAM Review 47*, 1 (2005), 50–66. doi:10.1137/S0036144503426074.

[11] BOLLOBÁS, B. *Random Graphs*, second ed., vol. 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 2001.

[12] BÖRGERS, C. *Mathematics of Social Choice*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2010. Voting, compensation, and division.

[13] BOTT, R., AND TU, L. W. *Differential Forms in Algebraic Topology*. Springer–Verlag, New York, 1982.

[14] BRANDT, A., MCCORMICK, S., AND RUGE, J. Algebraic multigrid (AMG) for sparse matrix equations. In *Sparsity and its applications (Loughborough, 1983)*. Cambridge Univ. Press, Cambridge, 1985, pp. 257–284.

[15] BROWN, M., AND SOKOL, J. An improved LRMC method for NCAA basketball prediction. *Journal of Quantitative Analysis in Sports 6*, 3 (2010).

[16] CALLAGHAN, T., MUCHA, P. J., AND PORTER, M. A. Random walker ranking for NCAA division I-A football. *Amer. Math. Monthly 114*, 9 (2007), 761–777.

[17] CHARTIER, T. P., KREUTZER, E., LANGVILLE, A. N., AND PEDINGS, K. E. Sensitivity and stability of ranking vectors. *SIAM Journal on Scientific Computing 33*, 3 (2011), 1077–1102. doi:DOI:10.1137/090772745.

[18] CHAVEL, I. *Eigenvalues in Riemannian Geometry*, vol. 115 of *Pure and Applied Mathematics*. Academic Press Inc., Orlando, FL, 1984. Including a chapter by Burton Randol, With an appendix by Jozef Dodziuk.

[19] CHORIN, A. J., AND MARSDEN, J. E. *A Mathematical Introduction to Fluid Mechanics*, third ed., vol. 4 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 1993.

[20] CHUNG, F., LU, L., AND VU, V. Spectra of random graphs with given expected degrees. *Proc. Natl. Acad. Sci. USA 100*, 11 (2003), 6313–6318. doi:10.1073/pnas.0937490100.

[21] CHUNG, F. R. K. *Spectral Graph Theory*, vol. 92 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 1997.

[22] DESBRUN, M., KANSO, E., AND TONG, Y. Discrete differential forms for computational modeling. In *Discrete Differential Geometry*, A. I. Bobenko, J. M. Sullivan, P. Schröder, and G. M. Ziegler, Eds., vol. 38 of *Oberwolfach Seminars*. Birkhäuser Basel, 2008, pp. 287–324. doi:10.1007/978-3-7643-8621-4_16.

[23] DEY, T. K., HIRANI, A. N., AND KRISHNAMOORTHY, B. Optimal homologous cycles, total unimodularity, and linear programming. *SIAM Journal on Computing 40*, 4 (2011), 1026–1044. doi:10.1137/100800245.

[24] ELO, A. *The Rating of Chess Players Past and Present*. Arco, 1978.

[25] ERDŐS, P., AND RÉNYI, A. On the evolution of random graphs. *Magyar Tud. Akad. Mat. Kutató Int. Közl. (Publ. Math. Inst. Hung. Acad. Sci.) 5* (1960), 17–61.

[26] FIEDLER, M. Algebraic connectivity of graphs. *Czechoslovak Math. J. 23(98)* (1973), 298–305.

[27] FRANKEL, T. *The Geometry of Physics*, second ed. Cambridge University Press, Cambridge, 2004. An introduction.

[28] GIRAULT, V., AND RAVIART, P.-A. *Finite Element Methods for Navier-Stokes Equations*. Springer–Verlag, 1986.

[29] GLEICH, D. F., AND LIM, L.-H. Rank aggregation via nuclear norm minimization, February 2011. Available as e-print on arxiv.org. arXiv:1102.4821v1.

[30] GLICKMAN, M. Parameter estimation in large dynamic paired comparison experiments. *Applied Statistics 48* (1999), 377–394.

[31] GRADY, L. J., AND POLIMENI, J. R. *Discrete Calculus*. Springer-Verlag London Ltd., London, 2010. Applied analysis on graphs for computational science. doi:10.1007/978-1-84996-290-2.

[32] GREENBAUM, A. *Iterative Methods for Solving Linear Systems*, vol. 17 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[33] HEATH, M. T. *Scientific Computing: An Introductory Survey*, second ed. McGraw-Hill, 2002.

[34] HIRANI, A. N. *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology, May 2003.

[35] JIANG, X., LIM, L.-H., YAO, Y., AND YE, Y. Statistical ranking and combinatorial hodge theory. *Mathematical Programming 127* (2011), 203–244. doi:10.1007/s10107-010-0419-x.

[36] JONES, E., OLIPHANT, T., AND PETERSON, P. SciPy: Open source scientific tools for Python, 2001.

[37] KAHLE, M. Topology of random clique complexes. *Discrete Math. 309*, 6 (2009), 1658–1671. doi:10.1016/j.disc.2008.02.037.

[38] KEENER, J. P. The Perron-Frobenius theorem and the ranking of football teams. *SIAM Rev. 35*, 1 (1993), 80–93. doi:10.1137/1035004.

[39] KLEINBERG, J. M. Authoritative sources in a hyperlinked environment. *J. ACM 46*, 5 (1999), 604–632. doi:10.1145/324133.324140.

[40] KOUTIS, I., MILLER, G., AND PENG, R. Approaching optimality for solving SDD systems. In *Proceedings of Foundations of Computer Science (to appear)* (2010).

[41] KVAM, P., AND SOKOL, J. S. A logistic regression/Markov chain model for NCAA basketball. *Naval Res. Logist. 53*, 8 (2006), 788–803. doi:10.1002/nav.20170.

[42] LEAKE, R. J. A method for ranking teams: With an application to college football. In *Management Science in Sports*, R. E. Machol and S. P. Ladany, Eds., vol. 4 of *TIMS Studies in the Management Sciences*. North-Holland Publishing Company, 1976, pp. 27–46.

[43] LI, J.-S., AND ZHANG, X.-D. On the Laplacian eigenvalues of a graph. *Linear Algebra Appl. 285*, 1-3 (1998), 305–307. doi:10.1016/S0024-3795(98)10149-0.

[44] MASSEY, K. Statistical models applied to the rating of sports teams. Honors project, Bluefield College, Bluefield, VA, 1997.

[45] MERRIS, R. A survey of graph Laplacians. *Linear and Multilinear Algebra 39*, 1-2 (1995), 19–31. doi:10.1080/03081089508818377.

[46] MOHAR, B. The Laplacian spectrum of graphs. In *Graph Theory, Combinatorics, and Applications. Vol. 2 (Kalamazoo, MI, 1988)*, Wiley-Intersci. Publ. Wiley, New York, 1991, pp. 871–898.

[47] MORITA, S. *Geometry of Differential Forms*, vol. 201 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 2001. Translated from the two-volume Japanese original (1997, 1998) by Teruko Nagase and Katsumi Nomizu, Iwanami Series in Modern Mathematics.

[48] MUNKRES, J. R. *Elements of Algebraic Topology*. Addison–Wesley Publishing Company, Menlo Park, 1984.

[49] NEWMAN, M. E. J., STROGATZ, S. H., AND WATTS, D. J. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev. E 64*, 2 (Jul 2001), 026118. doi:10.1103/PhysRevE.64.026118.

[50] PAGE, L., BRIN, S., MOTWANI, R., AND WINOGRAD, T. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[51] PAIGE, C. C., AND SAUNDERS, M. A. Algorithm 583: LSQR: Sparse linear equations and least squares problems. *ACM Trans. Math. Softw. 8*, 2 (1982), 195–209. doi:http://doi.acm.org/10.1145/355993.356000.

[52] PAIGE, C. C., AND SAUNDERS, M. A. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software 8*, 1 (Mar. 1982), 43–71.

[53] PARLETT, B. N. *The Symmetric Eigenvalue Problem*, vol. 20 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1998. Corrected reprint of the 1980 original.

[54] REYNOLDS, J. F. A proof of the random-walk method for solving Laplace's equation in 2-d. *The Mathematical Gazette 49*, 370 (1965), 416–420.

[55] RUGE, J. W., AND STÜBEN, K. Algebraic multigrid. In *Multigrid methods*, vol. 3 of *Frontiers Appl. Math.* SIAM, Philadelphia, PA, 1987, pp. 73–130.

[56] SAAD, Y. *Iterative Methods for Sparse Linear Systems*, second ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2003.

[57] STRANG, G. *Linear Algebra and its Applications*, third ed. Harcourt, Brace, Jovanovich, 1988.

[58] STÜBEN, K. A review of algebraic multigrid. *Journal of Computational and Applied Mathematics 128* (2001), 281–309. doi:10.1016/S0377-0427(00)00516-1.

[59] TIDEMAN, T. N. Independence of clones as a criterion for voting rules. *Soc. Choice Welf. 4*, 3 (1987), 185–206. doi:10.1007/BF00433944.

[60] TONG, Y., LOMBEYDA, S., HIRANI, A. N., AND DESBRUN, M. Discrete multiscale vector field decomposition. *ACM Transactions on Graphics (Special issue of SIGGRAPH 2003 Proceedings) 22*, 3 (July 2003), 445–452. doi:10.1145/882262.882290.

[61] TREFETHEN, L. N., AND BAU, III, D. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[62] VAN DER VORST, H. A. *Iterative Krylov Methods for Large Linear Systems*, vol. 13 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2009. Reprint of the 2003 original.

[63] VANĚK, P., MANDEL, J., AND BREZINA, M. Algebraic Multigrid by Smoothed Aggregation for Second and Fourth Order Elliptic Problems. *Computing 56*, 3 (1996), 179–196.

[64] YOUNG, H. P., AND LEVENGLICK, A. A consistent extension of Condorcet's election principle. *SIAM J. Appl. Math. 35*, 2 (1978), 285–300.

A Topology computations on clique complexes

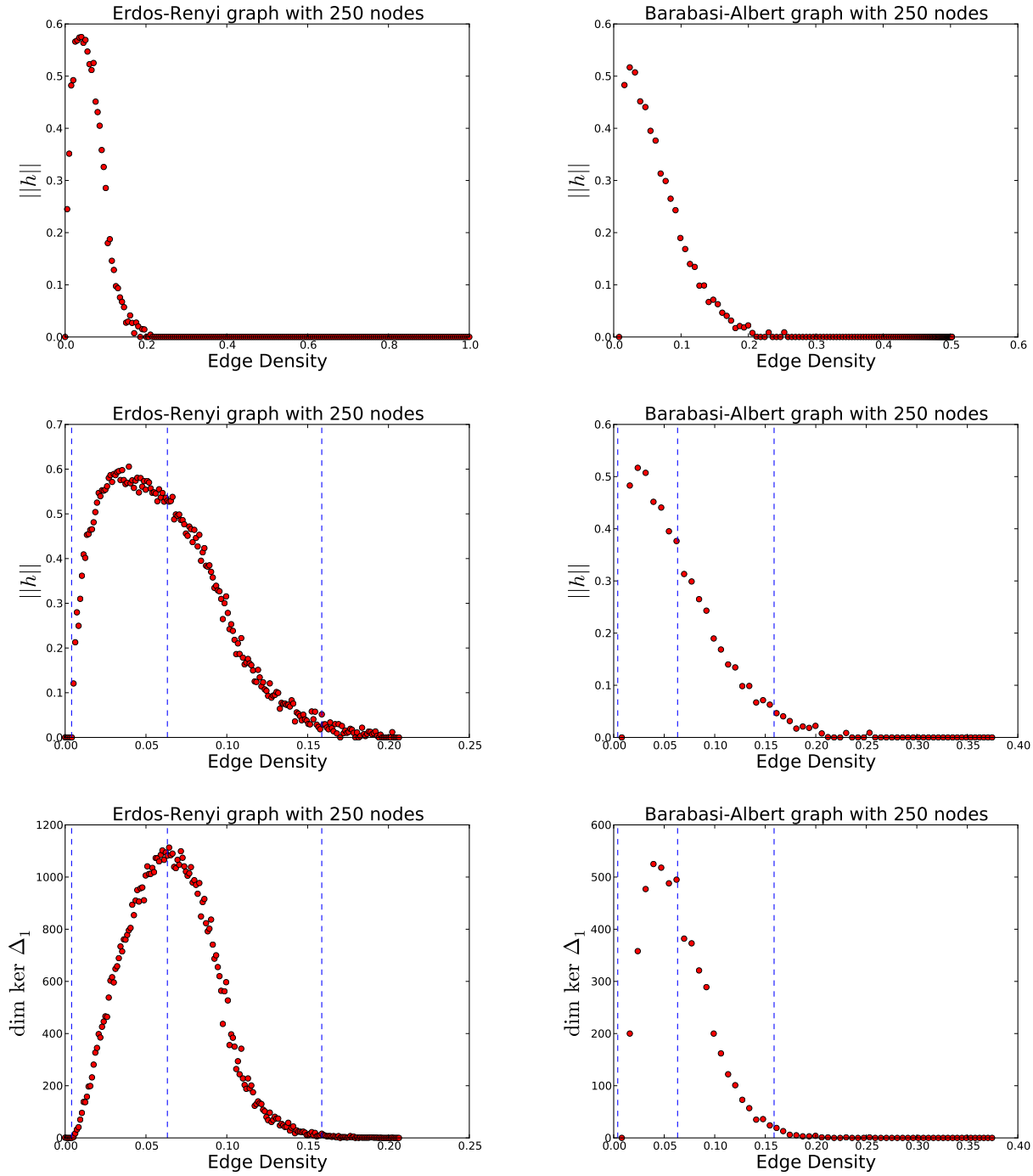


Figure 3: *Left column :* Erdős-Rényi random graphs; *Right column :* Barabási-Albert scale-free graphs. The top and middle rows show the norm of the harmonic component as a function of edge density. The bottom row shows the trends for the 1-dimensional Betti number. The bounds of Kahle [37] are the dashed vertical lines. They are also drawn for the right column graphs (although the theory is only for Erdős-Rényi graphs) to provide a comparison. See Section 7 for more details.

B Bounds on conjugate gradient iterations

Figures 4 and 5 show the conjugate gradient iteration estimates and actual number of iterations in experiments. The condition number of the Laplacian used is modulo the kernel, i.e., it is the ratio of the magnitude of the largest eigenvalue to the magnitude of the smallest nonzero eigenvalue. For certain special graphs, the condition number is explicitly known and the iteration count bound obtained from these is labeled "Bound from known cond(A)" in the figures. For Erdős-Rényi, Watts-Strogatz, and Barabási-Albert graphs there are known bounds on the condition number and the iteration count obtained from such bounds is denoted "Bound from bounded cond(A)". For such graphs we also measured the condition number (modulo kernel) to obtain iteration bounds that are labeled "Bound from exact cond(A)".

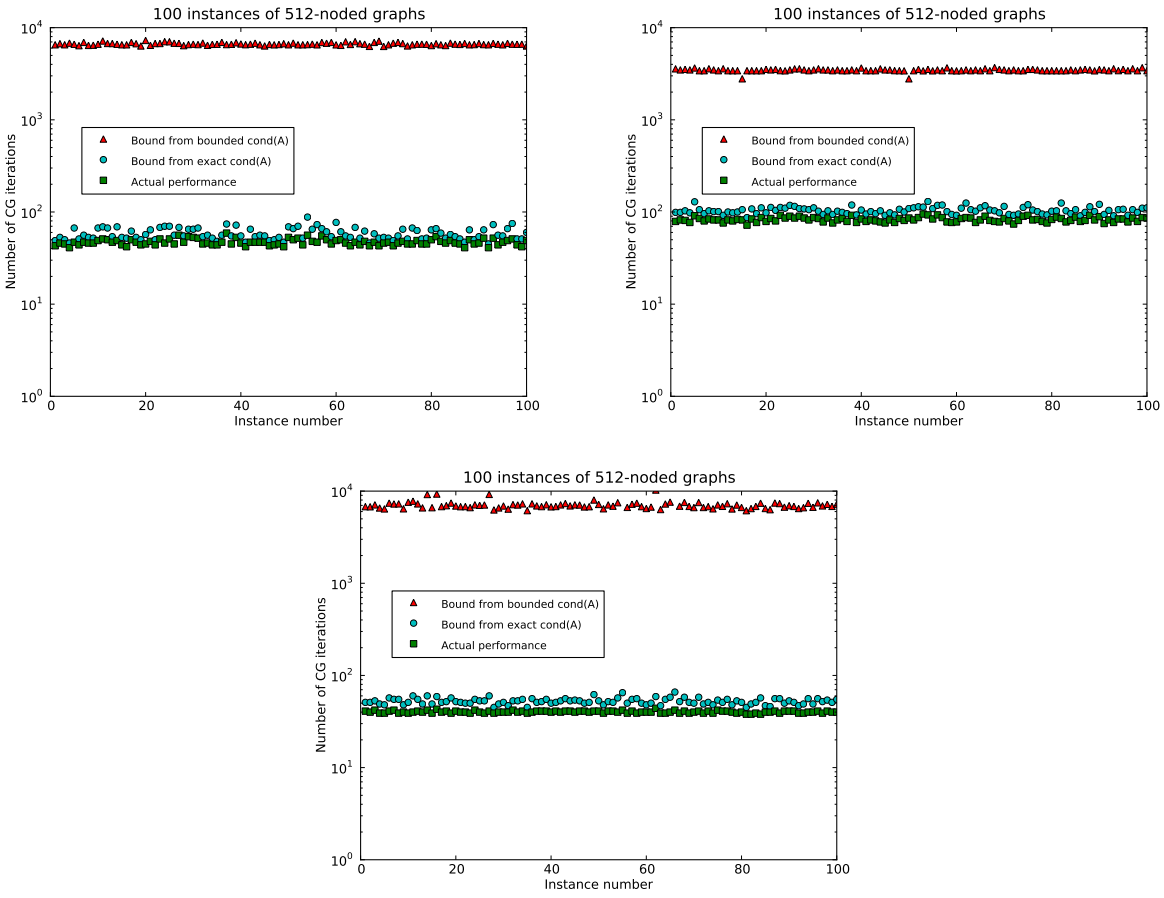


Figure 4: Conjugate gradient iteration bounds compared with actual number of iterations required to reduce the error $\|e_k\|_{\Delta_0}$ to 10^{-8} for a variety of graphs. *Top row :* results for Erdős-Rényi random graphs and Watts-Strogatz graphs. *Bottom row :* results for Barabási-Albert graphs.

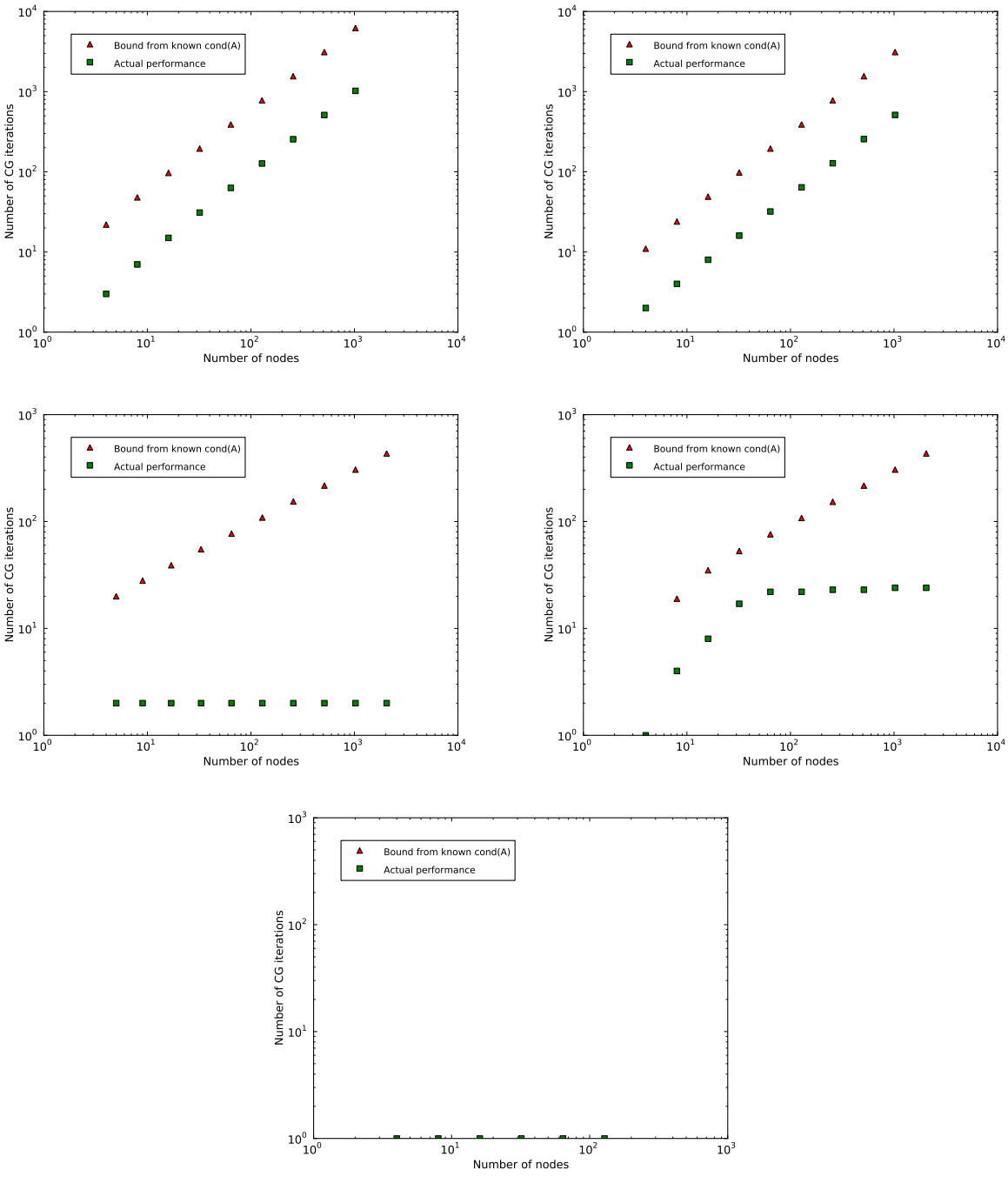


Figure 5: Conjugate gradient iteration bounds compared with actual number of iterations required to reduce the error $\|e_k\|_{\Delta_0}$ to 10^{-8} for a variety of graphs. *Top row :* results for path and cycle graphs. *Middle row :* results for star and wheel graphs. *Bottom row :* results for complete graphs. In the complete graph case, the markers for bounds are not visible in the plot because they coincide with the other markers.

C Timing and error using Krylov solvers

The next three tables show the results of numerical experiments on simplicial complexes Erdős-Rényi, Watts-Strogatz, and Barabási-Albert graphs. The cases where the homology of the simplicial 2-complex is trivial are marked by an asterisk $(*)$. In these cases it is the absolute error which is shown. See Section 8.3 for details.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	380	52	7.68e-02	3.22e-04	CG	1.1e-08	6.3e-09	2.0e-07	27	0.0041	17	0.0027
					MINRES	3.0e-08	1.3e-08	4.8e-07	26	0.0084	16	0.0053
					CG-K	1.0e-08	6.3e-09	1.3e-07	57	0.0090	35	0.0055
					MINRES-K	3.0e-08	1.3e-08	4.8e-07	53	0.0174	33	0.0108
					LSQR	2.8e-08	1.3e-08	4.7e-07	26	0.0166	16	0.0103
100	494	144	9.98e-02	8.91e-04	CG	1.1e-08	1.0e-08	2.9e-07	21	0.0037	30	0.0053
					MINRES	5.2e-08	1.1e-07	2.8e-06	19	0.0066	27	0.0095
					CG-K	4.8e-09	6.5e-09	1.5e-07	45	0.0082	438	0.0788
					MINRES-K	5.2e-08	5.3e-08	1.5e-06	39	0.0137	57	0.0201
					LSQR	4.9e-08	2.0e-08	8.9e-07	19	0.0121	29	0.0183
100	1212	2359	2.45e-01	1.46e-02	CG	3.9e-09	1.6e-08	3.5e-05	14	0.0023	60	0.0191
					MINRES	1.9e-08	1.1e-06	2.5e-03	13	0.0044	46	0.0229
					CG-K	3.9e-09	4.6e-09	1.0e-05	29	0.0053	129	0.0342
					MINRES-K	3.0e-07	2.7e-07	6.1e-04	23	0.0084	105	0.0466
					LSQR	7.6e-08	3.0e-08	7.6e-05	12	0.0082	58	0.0429
100	2530	21494	5.11e-01	1.33e-01	CG	2.5e-09	9.2e-09	3.0e-06*	10	0.0017	17	0.1033
					MINRES	2.0e-08	3.1e-07	1.0e-04*	9	0.0032	14	0.0897
					CG-K	1.9e-08	9.7e-09	2.1e-14*	19	0.0044	34	0.0438
					MINRES-K	1.0e-06	2.9e-07	2.1e-06*	15	0.0062	28	0.0476
					LSQR	1.3e-07	2.9e-09	2.9e-06*	8	0.0059	18	0.0362
100	3706	67865	7.49e-01	4.20e-01	CG	2.1e-09	6.6e-09	5.6e-06*	8	0.0015	11	0.4759
					MINRES	2.5e-08	7.5e-07	6.4e-04*	7	0.0026	8	0.3529
					CG-K	1.2e-07	6.8e-09	4.7e-14*	15	0.0041	22	0.1054
					MINRES-K	3.8e-06	7.4e-07	2.5e-06*	11	0.0050	16	0.0914
					LSQR	2.9e-07	2.3e-10	6.5e-06*	6	0.0047	13	0.0946
500	1290	21	1.03e-02	1.01e-06	CG	1.4e-08	1.9e-09	3.6e-07	43	0.0072	3	0.0007
					MINRES	1.6e-07	1.9e-09	4.0e-06	38	0.0129	3	0.0013
					CG-K	6.3e-09	1.9e-09	1.4e-07	93	0.0183	7	0.0013
					MINRES-K	6.6e-08	1.9e-09	1.7e-06	81	0.0302	7	0.0026
					LSQR	2.2e-08	1.9e-09	5.5e-07	42	0.0278	3	0.0023
500	12394	20315	9.94e-02	9.81e-04	CG	3.3e-09	2.7e-08	8.9e-05	13	0.0030	105	0.2155
					MINRES	7.2e-08	3.9e-06	1.3e-02	11	0.0045	72	0.1755
					CG-K	3.3e-09	8.0e-09	2.7e-05	27	0.0154	227	0.3955
					MINRES-K	3.4e-07	1.4e-06	4.7e-03	21	0.0161	161	0.3604
					LSQR	7.0e-08	5.6e-08	2.0e-04	11	0.0106	100	0.2348
500	24788	162986	1.99e-01	7.87e-03	CG	1.9e-09	1.3e-08	1.0e-05*	11	0.0032	25	1.447
					MINRES	6.8e-08	1.8e-06	1.4e-03*	9	0.0043	18	1.149
					CG-K	1.4e-08	7.5e-09	5.7e-14*	21	0.0303	52	0.7681
					MINRES-K	2.4e-06	1.6e-06	1.4e-04*	15	0.0257	37	0.8263
					LSQR	6.7e-08	3.3e-09	5.1e-06*	9	0.0123	27	0.6047
1000	49690	163767	9.95e-02	9.86e-04	CG	1.9e-09	1.4e-08	2.5e-03	11	0.0049	52	1.916
					MINRES	7.7e-08	4.2e-06	7.4e-01	9	0.0057	34	1.43
					CG-K	2.2e-09	4.5e-09	8.0e-04	23	0.0731	113	1.976
					MINRES-K	4.8e-07	1.1e-06	1.9e-01	17	0.0629	77	2.307
					LSQR	7.6e-08	1.4e-08	3.1e-03	9	0.0252	52	1.408

Table 1: Erdős-Rényi graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	500	729	1.01e-01	4.51e-03	CG	1.1e-08	1.3e-08	6.7e-07	28	0.0043	39	0.0075
					MINRES	3.8e-08	3.3e-07	1.6e-05	27	0.0088	33	0.0122
					CG-K	1.1e-08	7.4e-09	2.6e-07	57	0.0091	99	0.0182
					MINRES-K	4.7e-07	1.5e-07	9.5e-06	49	0.0164	71	0.0254
					LSQR	2.0e-07	2.3e-08	2.9e-06	25	0.0160	38	0.0251
100	1000	3655	2.02e-01	2.26e-02	CG	2.6e-09	1.6e-08	2.3e-06	18	0.0029	44	0.0252
					MINRES	1.5e-08	6.0e-07	9.0e-05	17	0.0057	35	0.0264
					CG-K	1.5e-08	4.3e-09	7.2e-07	35	0.0062	97	0.0295
					MINRES-K	1.7e-06	1.5e-07	4.2e-05	29	0.0103	79	0.0381
					LSQR	8.2e-08	1.6e-08	2.9e-06	16	0.0106	44	0.0346
100	1500	8354	3.03e-01	5.17e-02	CG	7.4e-09	1.7e-08	6.7e-05	13	0.0021	109	0.2024
					MINRES	7.4e-09	2.4e-06	9.6e-03	13	0.0044	84	0.1743
					CG-K	8.8e-09	7.2e-09	2.9e-05	27	0.0053	227	0.1091
					MINRES-K	2.9e-06	5.2e-07	2.3e-03	21	0.0079	185	0.1249
					LSQR	6.3e-08	1.7e-08	7.2e-05	12	0.0083	109	0.1067
100	2000	15530	4.04e-01	9.60e-02	CG	7.1e-09	2.0e-08	4.8e-06*	11	0.0019	68	0.2842
					MINRES	7.1e-09	2.9e-06	6.9e-04*	11	0.0038	53	0.2332
					CG-K	7.1e-09	8.4e-09	1.7e-14*	23	0.0049	142	0.1167
					MINRES-K	5.0e-06	4.7e-07	6.8e-05*	17	0.0067	118	0.1401
					LSQR	5.0e-08	6.1e-09	1.7e-06*	10	0.0071	71	0.0977
500	2500	3720	2.00e-02	1.80e-04	CG	1.8e-08	1.1e-08	1.2e-06	38	0.0066	44	0.0169
					MINRES	1.8e-07	6.3e-07	6.2e-05	33	0.0114	36	0.0203
					CG-K	1.8e-08	4.8e-09	6.9e-07	77	0.0184	93	0.0326
					MINRES-K	1.7e-06	2.2e-07	5.2e-05	57	0.0237	77	0.0410
					LSQR	2.3e-07	2.8e-08	7.0e-06	32	0.0222	42	0.0346
500	5000	16948	4.01e-02	8.18e-04	CG	1.2e-08	1.6e-08	4.5e-06	32	0.0059	34	0.0898
					MINRES	1.0e-07	6.2e-07	1.7e-04	28	0.0100	26	0.0771
					CG-K	2.2e-08	4.0e-09	1.4e-06	63	0.0201	75	0.0795
					MINRES-K	5.2e-06	2.0e-07	2.2e-04	43	0.0215	57	0.0841
					LSQR	4.7e-07	1.1e-08	1.9e-05	25	0.0187	35	0.0594
500	12500	110507	1.00e-01	5.34e-03	CG	8.4e-09	4.0e-08	4.5e-05	23	0.0051	161	6.469
					MINRES	2.0e-08	1.8e-05	2.1e-02	22	0.0088	86	3.632
					CG-K	1.9e-08	7.6e-09	8.5e-06	45	0.0254	367	3.337
					MINRES-K	1.3e-05	3.6e-06	4.1e-03	29	0.0223	217	2.617
					LSQR	9.5e-07	2.6e-08	7.4e-05	18	0.0176	166	1.996
1000	5000	7386	1.00e-02	4.44e-05	CG	1.4e-08	1.2e-08	1.8e-06	43	0.0085	44	0.0278
					MINRES	3.0e-07	7.5e-07	1.0e-04	36	0.0135	34	0.0280
					CG-K	2.2e-08	5.6e-09	1.1e-06	85	0.0284	93	0.0522
					MINRES-K	2.9e-06	3.4e-07	1.2e-04	61	0.0315	73	0.0554
					LSQR	3.6e-07	2.8e-08	1.4e-05	35	0.0268	42	0.0437
1000	10000	33022	2.00e-02	1.99e-04	CG	1.7e-08	1.4e-08	5.2e-06	32	0.0072	39	0.1953
					MINRES	2.1e-07	1.3e-06	4.8e-04	27	0.0108	28	0.1508
					CG-K	2.3e-08	5.0e-09	2.1e-06	65	0.0324	85	0.2098
					MINRES-K	6.7e-06	5.3e-07	4.2e-04	43	0.0297	63	0.1852
					LSQR	3.0e-07	1.4e-08	1.7e-05	26	0.0230	39	0.1365
1000	25000	220002	5.01e-02	1.32e-03	CG	1.0e-08	2.4e-08	3.4e-05	27	0.0080	85	7.04
					MINRES	2.1e-07	1.2e-05	1.8e-02	23	0.0110	48	4.52
					CG-K	2.4e-08	4.6e-09	6.9e-06	53	0.0688	189	3.7
					MINRES-K	2.6e-05	2.3e-06	4.0e-03	31	0.0554	119	3.588
					LSQR	5.9e-07	1.2e-08	5.5e-05	21	0.0383	89	2.712

Table 2: Watts-Strogatz graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	475	301	9.60e-02	1.86e-03	CG	1.1e-08	2.4e-08	1.3e-06	25	0.0038	79	0.0127
					MINRES	2.7e-08	3.2e-07	1.8e-05	24	0.0078	70	0.0233
					CG-K	8.3e-09	7.3e-09	4.1e-07	55	0.0088	171	0.0282
					MINRES-K	1.6e-07	2.0e-07	1.2e-05	45	0.0149	145	0.0497
					LSQR	1.5e-07	7.0e-08	4.9e-06	22	0.0142	75	0.0476
100	900	1701	1.82e-01	1.05e-02	CG	1.1e-08	2.6e-08	1.0e-05	21	0.0033	96	0.0281
					MINRES	3.3e-08	1.1e-06	4.2e-04	20	0.0067	74	0.0351
					CG-K	1.5e-08	7.5e-09	2.6e-06	45	0.0079	211	0.0494
					MINRES-K	6.0e-07	3.8e-07	1.6e-04	35	0.0123	165	0.0682
					LSQR	2.4e-07	4.6e-08	2.5e-05	18	0.0119	93	0.0653
100	1600	8105	3.23e-01	5.01e-02	CG	6.7e-09	1.3e-08	2.2e-06*	20	0.0032	50	0.0967
					MINRES	2.0e-08	7.8e-07	1.3e-04*	19	0.0063	39	0.0832
					CG-K	2.3e-08	8.1e-09	9.3e-11*	39	0.0077	106	0.0506
					MINRES-K	1.4e-06	3.4e-07	1.2e-05*	31	0.0116	82	0.0545
					LSQR	4.7e-07	9.3e-09	7.0e-06*	16	0.0108	51	0.0499
100	2400	24497	4.85e-01	1.51e-01	CG	5.2e-09	1.2e-08	4.3e-06*	18	0.0030	27	0.2435
					MINRES	5.0e-08	6.8e-07	2.4e-04*	16	0.0054	21	0.1960
					CG-K	1.8e-08	7.4e-09	2.2e-14*	35	0.0078	56	0.0817
					MINRES-K	4.6e-06	3.1e-07	6.1e-06*	25	0.0100	44	0.0831
					LSQR	4.2e-07	3.2e-09	8.0e-06*	14	0.0097	29	0.0632
500	4900	4740	3.93e-02	2.29e-04	CG	1.0e-08	3.4e-08	9.1e-06	32	0.0059	165	0.0956
					MINRES	4.7e-08	1.8e-06	4.8e-04	30	0.0107	118	0.0904
					CG-K	2.2e-08	8.2e-09	2.2e-06	73	0.0231	367	0.1707
					MINRES-K	6.8e-07	8.2e-07	2.2e-04	53	0.0262	257	0.1669
					LSQR	3.1e-07	1.2e-07	3.8e-05	27	0.0202	150	0.1378
500	9600	25016	7.70e-02	1.21e-03	CG	8.4e-09	3.0e-08	5.8e-05	27	0.0056	215	1.043
					MINRES	8.2e-08	4.7e-06	9.1e-03	24	0.0092	133	0.6919
					CG-K	1.1e-08	8.5e-09	1.3e-05	57	0.0267	477	0.9132
					MINRES-K	1.5e-06	1.7e-06	3.3e-03	41	0.0270	303	0.7096
					LSQR	3.5e-07	7.9e-08	1.8e-04	22	0.0191	202	0.5085
500	18400	133933	1.47e-01	6.47e-03	CG	4.5e-09	2.2e-08	2.0e-03	23	0.0058	111	7.204
					MINRES	6.8e-08	5.8e-06	5.3e-01	20	0.0086	69	4.667
					CG-K	1.8e-08	3.8e-09	3.6e-04	45	0.0414	251	2.809
					MINRES-K	1.1e-05	1.2e-06	1.3e-01	29	0.0362	163	2.646
					LSQR	3.4e-07	1.7e-08	2.5e-03	18	0.0205	113	1.867
1000	9900	6264	1.98e-02	3.77e-05	CG	7.4e-09	4.4e-08	1.1e-05	41	0.0091	211	0.1410
					MINRES	8.5e-08	3.2e-06	8.1e-04	37	0.0147	146	0.1246
					CG-K	1.4e-08	1.1e-08	2.8e-06	89	0.0442	481	0.3244
					MINRES-K	4.5e-07	1.1e-06	2.9e-04	69	0.0474	321	0.2883
					LSQR	2.3e-07	1.8e-07	5.0e-05	35	0.0309	185	0.2043
1000	19600	37365	3.92e-02	2.25e-04	CG	9.0e-09	4.1e-08	5.8e-05	33	0.0088	335	2.521
					MINRES	7.5e-08	1.0e-05	1.4e-02	30	0.0133	200	1.594
					CG-K	1.3e-08	8.3e-09	9.7e-06	69	0.0641	769	2.833
					MINRES-K	2.8e-06	2.9e-06	4.1e-03	49	0.0658	469	1.922
					LSQR	4.0e-07	1.3e-07	2.0e-04	27	0.0363	309	1.237
1000	38400	202731	7.69e-02	1.22e-03	CG	5.7e-09	3.0e-08	5.2e-04	27	0.0096	211	20.65
					MINRES	5.5e-08	1.3e-05	2.2e-01	24	0.0127	120	13.12
					CG-K	4.2e-08	5.4e-09	8.3e-05	55	0.1219	479	9.466
					MINRES-K	9.9e-06	2.8e-06	5.0e-02	35	0.0943	289	9.651
					LSQR	4.8e-07	3.8e-08	8.9e-04	21	0.0421	208	6.108

Table 3: Barabási-Albert graphs.

D Timing and error using algebraic multigrid

The next three tables show the results of numerical experiments using algebraic multigrid on simplicial complexes generated from Erdős-Rényi, Watts-Strogatz, and Barabási-Albert graphs. We used the PyAMG implementation of algebraic multigrid [9]. As in the Krylov tables earlier, the cases where the homology of the simplicial 2-complex is trivial are marked by an asterisk (∗). The cases where PyAMG failed in the setup phase are marked by a dagger symbol (†), and the cases where algebraic multigrid reached maximum number of specified iterations are marked by the double dagger symbol (‡). In Table 4 the entries marked with a dash symbol (–) are those for which Algorithm 1 (Schur complement and algebraic multigrid algorithm, see Section 8.5) cannot be applied because Δ_2 has no simple sparse/dense partitioning. See Section 8.4 for more details.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	380	52	7.68e-02	3.22e-04	AMG (SA)	5.0e-09	3.8e-09	9.5e-08	2	0.0577	2	0.0030
					AMG (LA)	5.0e-09	3.8e-09	9.5e-08	2	0.0078	2	0.0023
					Schur	5.0e-09	–	–	3	0.0430	–	–
100	494	144	9.98e-02	8.91e-04	AMG (SA)	2.8e-09	3.8e-09	1.0e-07	2	0.0078	2	0.0137
					AMG (LA)	2.8e-09	3.8e-09	1.0e-07	2	0.0082	2	0.0131
					Schur	2.8e-09	–	–	3	0.0228	–	–
100	1212	2359	2.45e-01	1.46e-02	AMG (SA)	1.2e-10	4.8e-08	1.1e-04	2	0.0079	32	0.6259
					AMG (LA)	1.2e-10	3.4e-08	7.5e-05	2	0.0079	30	0.1426
					Schur	1.2e-10	–	–	3	0.0194	–	–
100	2530	21494	5.11e-01	1.33e-01	AMG (SA)	6.5e-10	6.9e-09	2.3e-06*	2	0.0142	27	2.102
					AMG (LA)	6.5e-10	4.8e-09	1.6e-06*	2	0.0184	19	15.89
					Schur	6.5e-10	–	–	3	0.0185	–	–
100	3706	67865	7.49e-01	4.20e-01	AMG (SA)	2.8e-10	7.7e-09	6.6e-06*	2	0.0689	39	16.74
					AMG (LA)	2.8e-10	7.3e-09	6.2e-06*	2	0.0356	14	408.5
					Schur	2.8e-10	–	–	3	0.3740	–	–
500	1290	21	1.03e-02	1.01e-06	AMG (SA)	3.8e-09	1.9e-09	9.8e-08	2	0.7113	2	0.0360
					AMG (LA)	3.8e-09	1.9e-09	9.8e-08	2	0.3080	2	0.0022
					Schur	3.8e-09	–	–	3	0.2952	–	–
500	12394	20315	9.94e-02	9.81e-04	AMG (SA)	7.2e-11	7.7e-08	2.5e-04	2	0.3305	74	2.711
					AMG (LA)	7.2e-11	7.4e-08	2.5e-04	2	0.3293	71	8.358
					Schur	7.2e-11	–	–	3	0.2836	–	–
500	24788	162986	1.99e-01	7.87e-03	AMG (SA)	†	†	†	†	†	†	†
					AMG (LA)	†	†	†	†	†	†	†
					Schur	†	–	–	†	†	–	–
1000	49690	163767	9.95e-02	9.86e-04	AMG (SA)	†	†	†	†	†	†	†
					AMG (LA)	†	†	†	†	†	†	†
					Schur	†	–	–	†	†	–	–

Table 4: Erdős-Rényi graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	500	729	1.01e-01	4.51e-03	AMG (SA)	2.9e-09	2.4e-08	1.2e-06	2	0.0179	20	0.0577
					AMG (LA)	2.9e-09	1.4e-08	6.9e-07	2	0.0009	18	0.0324
100	1000	3655	2.02e-01	2.26e-02	AMG (SA)	1.7e-09	3.1e-08	4.6e-06	2	0.0009	23	0.1253
					AMG (LA)	1.7e-09	2.7e-08	4.0e-06	2	0.0009	17	0.1601
100	1500	8354	3.03e-01	5.17e-02	AMG (SA)	1.3e-10	7.6e-08	3.0e-04	2	0.0011	99	1.377
					AMG (LA)	1.3e-10	7.1e-08	2.8e-04	2	0.0011	60	2.345
100	2000	15530	4.04e-01	9.60e-02	AMG (SA)	3.8e-09	4.6e-08	1.1e-05*	2	0.0015	28	1.282
					AMG (LA)	3.8e-09	1.8e-08	4.3e-06*	2	0.0013	15	5.039
500	2500	3720	2.00e-02	1.80e-04	AMG (SA)	2.5e-09	3.0e-08	3.0e-06	2	0.0149	15	0.0650
					AMG (LA)	2.5e-09	4.4e-08	4.3e-06	2	0.0136	21	0.0929
500	5000	16948	4.01e-02	8.18e-04	AMG (SA)	6.0e-10	6.9e-09	1.9e-06	2	0.0139	14	0.4410
					AMG (LA)	6.0e-10	3.9e-09	1.1e-06	2	0.0140	12	0.7969
500	12500	110507	1.00e-01	5.34e-03	AMG (SA)	8.8e-10	3.9e-07	4.4e-04	2	0.0143	101‡	32.75
					AMG (LA)	8.8e-10	7.3e-08	8.2e-05	2	0.0145	86	173.1
1000	5000	7386	1.00e-02	4.44e-05	AMG (SA)	2.7e-08	2.7e-08	3.8e-06	14	0.0651	20	0.1325
					AMG (LA)	2.0e-08	2.5e-08	3.5e-06	17	0.0646	19	0.1715
1000	10000	33022	2.00e-02	1.99e-04	AMG (SA)	1.8e-08	1.6e-08	6.0e-06	17	0.0697	17	1.058
					AMG (LA)	2.4e-08	1.3e-08	5.1e-06	10	0.0715	13	1.798
1000	25000	220002	5.01e-02	1.32e-03	AMG (SA)	1.6e-08	2.9e-08	4.1e-05	15	0.0823	29	26.71
					AMG (LA)	5.2e-09	5.3e-08	7.6e-05	6	0.1034	23	241.3

Table 5: Watts-Strogatz graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	Relative Error $\|\partial_1^T \alpha\|$	Relative Error $\|\partial_2 \beta\|$	Relative Error $\|h\|$	α iter.	α sec.	β iter.	β sec.
100	475	301	9.60e-02	1.86e-03	AMG (SA)	3.2e-09	4.3e-09	2.4e-07	2	0.0009	2	0.0038
					AMG (LA)	3.2e-09	4.3e-09	2.4e-07	2	0.0011	2	0.0037
100	900	1701	1.82e-01	1.05e-02	AMG (SA)	6.7e-10	4.7e-08	1.8e-05	2	0.0016	50	0.0995
					AMG (LA)	6.7e-10	5.6e-08	2.2e-05	2	0.0009	44	0.1050
100	1600	8105	3.23e-01	5.01e-02	AMG (SA)	2.0e-09	1.2e-08	1.9e-06*	2	0.0009	14	0.3561
					AMG (LA)	2.0e-09	5.9e-09	9.8e-07*	2	0.0010	12	1.545
100	2400	24497	4.85e-01	1.51e-01	AMG (SA)	1.1e-09	7.3e-09	2.6e-06*	2	0.0017	24	2.632
					AMG (LA)	1.1e-09	7.6e-09	2.7e-06*	2	0.0013	13	22.41
500	4900	4740	3.93e-02	2.29e-04	AMG (SA)	2.3e-09	7.5e-08	2.0e-05	2	0.0169	89	0.4003
					AMG (LA)	2.3e-09	8.9e-08	2.4e-05	2	0.0137	95	0.5046
500	9600	25016	7.70e-02	1.21e-03	AMG (SA)	4.9e-10	7.3e-07	1.4e-03	2	0.0142	101‡	4.627
					AMG (LA)	4.9e-10	4.0e-07	7.8e-04	2	0.0142	101‡	29.74
500	18400	133933	1.47e-01	6.47e-03	AMG (SA)	†	†	†	†	†	†	†
					AMG (LA)	†	†	†	†	†	†	†
1000	9900	6264	1.98e-02	3.77e-05	AMG (SA)	5.3e-09	5.6e-06	1.4e-03	5	0.0477	101‡	0.5723
					AMG (LA)	3.5e-09	9.1e-06	2.3e-03	5	0.0682	101‡	1.22
1000	19600	37365	3.92e-02	2.25e-04	AMG (SA)	7.5e-09	1.1e-05	1.5e-02	4	0.0393	101‡	10.61
					AMG (LA)	5.6e-09	8.6e-06	1.2e-02	4	0.0823	101‡	63.01
1000	38400	202731	7.69e-02	1.22e-03	AMG (SA)	†	†	†	†	†	†	†
					AMG (LA)	†	†	†	†	†	†	†

Table 6: Barabási-Albert graphs.

E Setup and solve timings for algebraic multigrid

The setup phase of algebraic multigrid involves some precomputation. In the next three tables we separate the timing results shown in Tables 4–6 into the time required for the setup and solve phases.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	α iter.	α setup	α solve	α total	β iter.	β setup	β solve	β total
100	380	52	7.68e-02	3.22e-04	SA	2	0.0001	0.0576	0.0577	2	0.0001	0.0029	0.0030
					LA	2	0.0001	0.0077	0.0078	2	0.0001	0.0022	0.0023
100	494	144	9.98e-02	8.91e-04	SA	2	0.0001	0.0076	0.0078	2	0.0001	0.0136	0.0137
					LA	2	0.0001	0.0080	0.0082	2	0.0001	0.0129	0.0131
100	1212	2359	2.45e-01	1.46e-02	SA	2	0.0001	0.0078	0.0079	32	0.5708	0.0551	0.6259
					LA	2	0.0002	0.0077	0.0079	30	0.0392	0.1034	0.1426
100	2530	21494	5.11e-01	1.33e-01	SA	2	0.0030	0.0111	0.0142	27	0.8655	1.236	2.102
					LA	2	0.0048	0.0136	0.0184	19	12.36	3.528	15.89
100	3706	67865	7.49e-01	4.20e-01	SA	2	0.0303	0.0386	0.0689	39	5.652	11.08	16.74
					LA	2	0.0138	0.0219	0.0356	14	375.7	32.85	408.5
500	1290	21	1.03e-02	1.01e-06	SA	2	0.1353	0.5760	0.7113	2	0.0071	0.0289	0.0360
					LA	2	0.0001	0.3079	0.3080	2	0.0007	0.0015	0.0022
500	12394	20315	9.94e-02	9.81e-04	SA	2	0.0001	0.3303	0.3305	74	0.4907	2.22	2.711
					LA	2	0.0041	0.3252	0.3293	71	3.082	5.277	8.358

Table 7: Erdős-Rényi graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	α iter.	α setup	α solve	α total	β iter.	β setup	β solve	β total
100	500	729	1.01e-01	4.51e-03	SA	2	0.0001	0.0178	0.0179	20	0.0410	0.0167	0.0577
					LA	2	0.0001	0.0008	0.0009	18	0.0172	0.0152	0.0324
100	1000	3655	2.02e-01	2.26e-02	SA	2	0.0001	0.0008	0.0009	23	0.0553	0.0700	0.1253
					LA	2	0.0001	0.0008	0.0009	17	0.0939	0.0663	0.1601
100	1500	8354	3.03e-01	5.17e-02	SA	2	0.0002	0.0009	0.0011	99	0.1754	1.202	1.377
					LA	2	0.0002	0.0009	0.0011	60	0.7271	1.618	2.345
100	2000	15530	4.04e-01	9.60e-02	SA	2	0.0004	0.0011	0.0015	28	0.4623	0.8194	1.282
					LA	2	0.0003	0.0011	0.0013	15	3.793	1.246	5.039
500	2500	3720	2.00e-02	1.80e-04	SA	2	0.0007	0.0142	0.0149	15	0.0360	0.0290	0.0650
					LA	2	0.0001	0.0135	0.0136	21	0.0432	0.0497	0.0929
500	5000	16948	4.01e-02	8.18e-04	SA	2	0.0001	0.0138	0.0139	14	0.2127	0.2283	0.4410
					LA	2	0.0002	0.0138	0.0140	12	0.5014	0.2955	0.7969
500	12500	110507	1.00e-01	5.34e-03	SA	2	0.0003	0.0140	0.0143	101	4.96	27.79	32.75
					LA	2	0.0004	0.0141	0.0145	86	86.51	86.57	173.1
1000	5000	7386	1.00e-02	4.44e-05	SA	14	0.0266	0.0385	0.0651	20	0.0577	0.0748	0.1325
					LA	17	0.0250	0.0396	0.0646	19	0.0884	0.0831	0.1715
1000	10000	33022	2.00e-02	1.99e-04	SA	17	0.0264	0.0433	0.0697	17	0.4924	0.5655	1.058
					LA	10	0.0307	0.0407	0.0715	13	1.152	0.6456	1.798
1000	25000	220002	5.01e-02	1.32e-03	SA	15	0.0308	0.0515	0.0823	29	10.21	16.5	26.71
					LA	6	0.0474	0.0559	0.1034	23	192.5	48.85	241.3

Table 8: Watts-Strogatz graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Algorithm / Formulation	α iter.	α setup	α solve	α total	β iter.	β setup	β solve	β total
100	475	301	9.60e-02	1.86e-03	SA	2	0.0001	0.0008	0.0009	2	0.0001	0.0037	0.0038
					LA	2	0.0002	0.0009	0.0011	2	0.0001	0.0037	0.0037
100	900	1701	1.82e-01	1.05e-02	SA	2	0.0005	0.0012	0.0016	50	0.0242	0.0753	0.0995
					LA	2	0.0001	0.0008	0.0009	44	0.0358	0.0692	0.1050
100	1600	8105	3.23e-01	5.01e-02	SA	2	0.0001	0.0008	0.0009	14	0.1842	0.1720	0.3561
					LA	2	0.0001	0.0009	0.0010	12	1.149	0.3968	1.545
100	2400	24497	4.85e-01	1.51e-01	SA	2	0.0005	0.0012	0.0017	24	1.111	1.521	2.632
					LA	2	0.0002	0.0011	0.0013	13	19.39	3.023	22.41
500	4900	4740	3.93e-02	2.29e-04	SA	2	0.0016	0.0153	0.0169	89	0.0742	0.3261	0.4003
					LA	2	0.0001	0.0136	0.0137	95	0.1027	0.4019	0.5046
500	9600	25016	7.70e-02	1.21e-03	SA	2	0.0002	0.0140	0.0142	101	1.018	3.608	4.627
					LA	2	0.0004	0.0138	0.0142	101	10.56	19.19	29.74
1000	9900	6264	1.98e-02	3.77e-05	SA	5	0.0215	0.0261	0.0477	101	0.0946	0.4777	0.5723
					LA	5	0.0316	0.0366	0.0682	101	0.2080	1.012	1.22
1000	19600	37365	3.92e-02	2.25e-04	SA	4	0.0174	0.0219	0.0393	101	2.403	8.212	10.61
					LA	4	0.0387	0.0436	0.0823	101	27.14	35.86	63.01

Table 9: Barabási-Albert graphs.

F Nonzeros in Laplacian and boundary matrices

The next three tables show the number of nonzeros in the boundary and Laplacian matrices. In addition we also show the number of nonzeros in the coarse grid matrices that arise in algebraic multigrid. The latter can be useful in interpreting and debugging algebraic multigrid performance. The cases where PyAMG failed in the setup phase are marked by a dagger symbol (†).

N_0	N_1	N_2	Edge Density	Triangle Density	Krylov				AMG			
									SA		LA	
					∂_1	∂_2	Δ_0	Δ_2	Δ_0	Δ_2	Δ_0	Δ_2
100	380	52	7.68e-02	3.22e-04	760	156	860	110	860	110	860	110
100	494	144	9.98e-02	8.91e-04	988	432	1088	528	1088	528	1088	528
100	1212	2359	2.45e-01	1.46e-02	2424	7077	2524	42475	2524	42475 3721	2524	42475 49902
100	2530	21494	5.11e-01	1.33e-01	5060	64482	5160	1645012	5160	1645012 2500	5160	1645012 4174983 45796
100	3706	67865	7.49e-01	4.20e-01	7412	203595	7512	11134203	7512	11134203 1681	7512	11134203 34985035 459684 4489
500	1290	21	1.03e-02	1.01e-06	2580	63	3075	25	3075	25	3075	25
500	12394	20315	9.94e-02	9.81e-04	24788	60945	25288	319503	25288	319503 454789 1	25288	319503 1739567 41209
500	24788	162986	1.99e-01	7.87e-03	49576	488958	50076	9807176	†	†	†	†
1000	49690	163767	9.95e-02	9.86e-04	99380	491301	100380	5033205	†	†	†	†

Table 10: Erdős-Rényi graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Krylov				AMG			
									SA		LA	
					∂_1	∂_2	Δ_0	Δ_2	Δ_0	Δ_2	Δ_0	Δ_2
100	500	729	1.01e-01	4.51e-03	1000	2187	1100	10053	1100	10053 302	1100	10053 1145
100	1000	3655	2.02e-01	2.26e-02	2000	10965	2100	127001	2100	127001 929	2100	127001 50045
100	1500	8354	3.03e-01	5.17e-02	3000	25062	3100	443774	3100	443774 1921	3100	443774 373960 6889
100	2000	15530	4.04e-01	9.60e-02	4000	46590	4100	1138022	4100	1138022 2116	4100	1138022 1622123 24025
500	2500	3720	2.00e-02	1.80e-04	5000	11160	5500	52456	5500	52456 5500	5500	52456 6210
500	5000	16948	4.01e-02	8.18e-04	10000	50844	10500	568252	5500	568252 3808	10500	568252 205516 10015
500	12500	110507	1.00e-01	5.34e-03	25000	331521	25500	9870149	25500	9870149 38306	25500	9870149 16959388 1221025 12100
1000	5000	7386	1.00e-02	4.44e-05	10000	22158	11000	103946	11000 5231	103946 2787	11000 8110	103946 12339 631
1000	10000	33022	2.00e-02	1.99e-04	20000	99066	21000	1086816	21000 900	1086816 7442	21000 10000	1086816 393322 22318
1000	25000	220002	5.01e-02	1.32e-03	50000	660006	51000	19688290	51000 121	19688290 42253	51000 10000	19688290 32579049 4819514 48400

Table 11: Watts-Strogatz graphs.

N_0	N_1	N_2	Edge Density	Triangle Density	Krylov				AMG			
									SA		LA	
					∂_1	∂_2	Δ_0	Δ_2	Δ_0	Δ_2	Δ_0	Δ_2
100	475	301	9.60e-02	1.86e-03	950	903	1050	3189	1050	3189	1050	3189
100	900	1701	1.82e-01	1.05e-02	1800	5103	1900	42465	1900	42465 1837	1900	42465 26324
100	1600	8105	3.23e-01	5.01e-02	3200	24315	3300	472503	3300	472503 2401	3300	472503 607290 6561
100	2400	24497	4.85e-01	1.51e-01	4800	73491	4900	2610185	4900	2610185 1936	4900	2610185 5244756 59536
500	4900	4740	3.93e-02	2.29e-04	9800	14220	10300	121426	10300	121426 44271	10300	121426 138744
500	9600	25016	7.70e-02	1.21e-03	19200	75048	19700	1180338	19700	1180338 161347	19700	1180338 4353167 62500
500	18400	133933	1.47e-01	6.47e-03	36800	401799	37300	14303959	†	†	†	†
1000	9900	6264	1.98e-02	3.77e-05	19800	18792	20800	132712	20800 196	132712 119206	20800 10000	132712 161391 3722
1000	19600	37365	3.92e-02	2.25e-04	39200	112095	40200	1808827	40200 16	1808827 774769 1	40200 10000	1808827 8082839 139129
1000	38400	202731	7.69e-02	1.22e-03	76800	608193	77800	19768919	†	†	†	†

Table 12: Barabási-Albert graphs.

G Nonzero patterns in Laplacians

The next nine figures show the nonzero patterns for the Laplacian matrices Δ_0 and Δ_2 and their various reorderings for different types of graphs. The four rows from top to bottom in some of the figures show the pattern of nonzeros in the Δ_0 or Δ_2 matrix and their reordering. The reorderings are by vertex degree (or number of nonzeros), reverse Cuthill-McKee algorithm, and Sloan's algorithm. The rows are labelled as "Original", "Degree" (or "NNZ"), "RCMK", and "Sloan". In each column the size of matrix is shown in the top row. In each collection all matrices are drawn with the same size. This results in some visual artifacts in the Δ_2 figures where the matrices are of vastly different sizes. For example, in the middle column of Figure 8 the top matrix and the bottom matrix have the same number of nonzeros even though it does not appear that way in the figure.

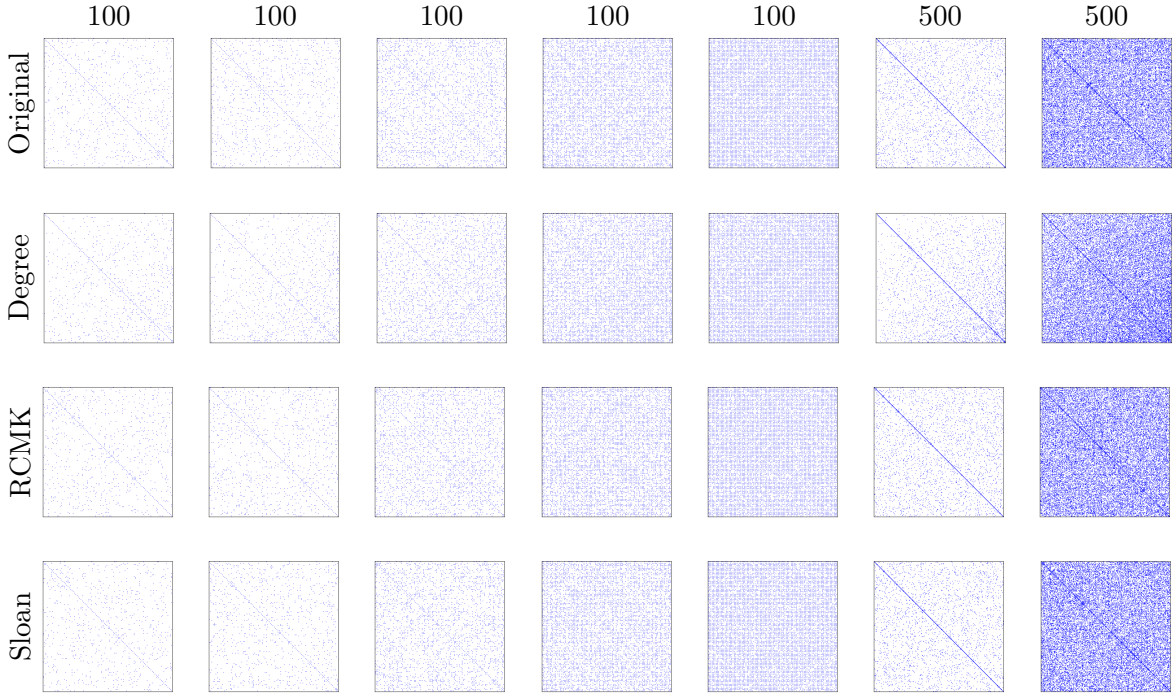


Figure 6: Δ_0 for Erdős-Rényi graphs.

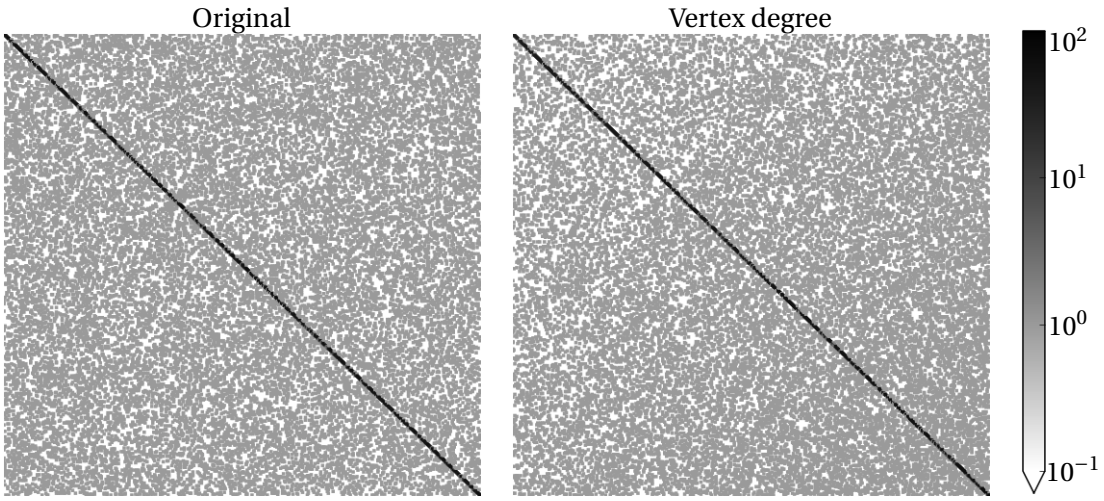


Figure 7: Enlarged view of the matrix corresponding to the last column in Figure 6.

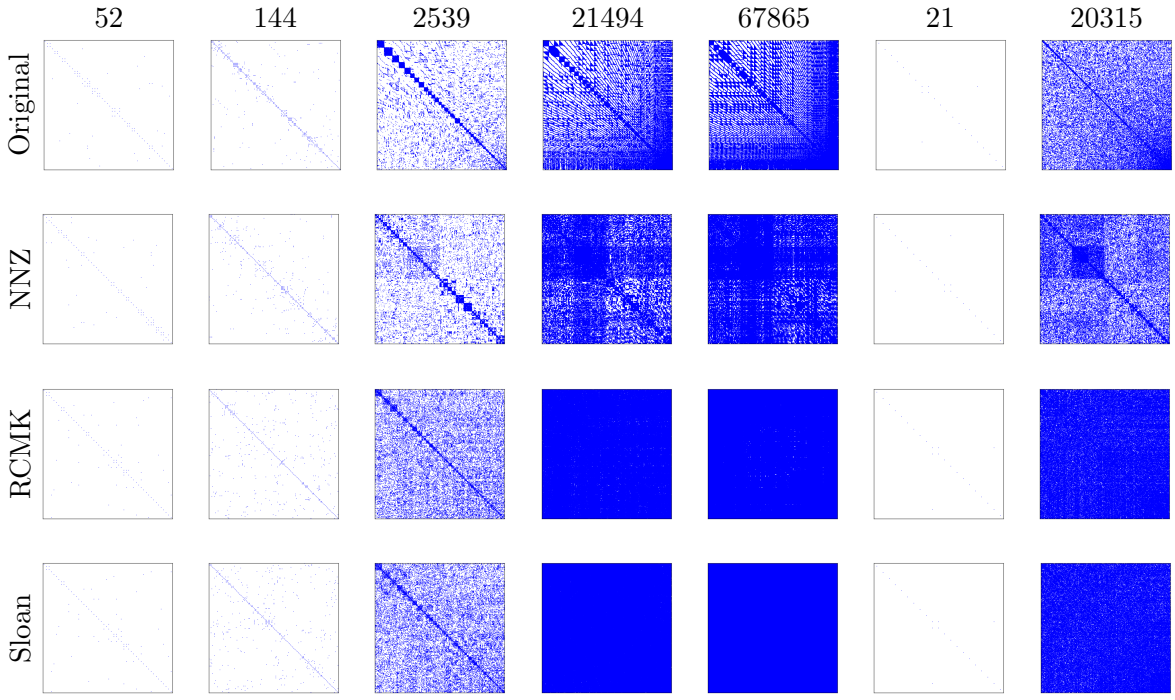


Figure 8: Δ_2 for Erdős-Rényi graphs.

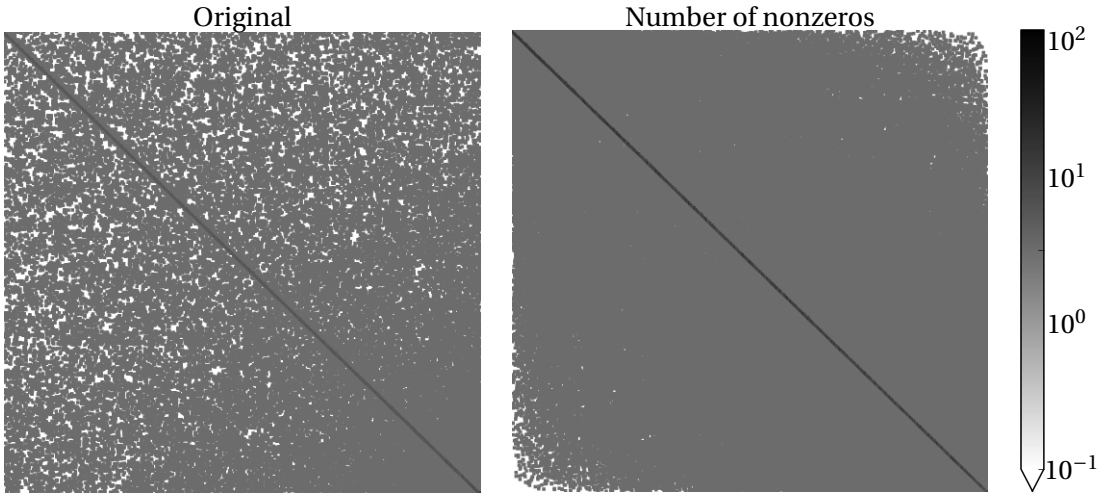


Figure 9: Enlarged view of the matrix corresponding to the last column in Figure 8.

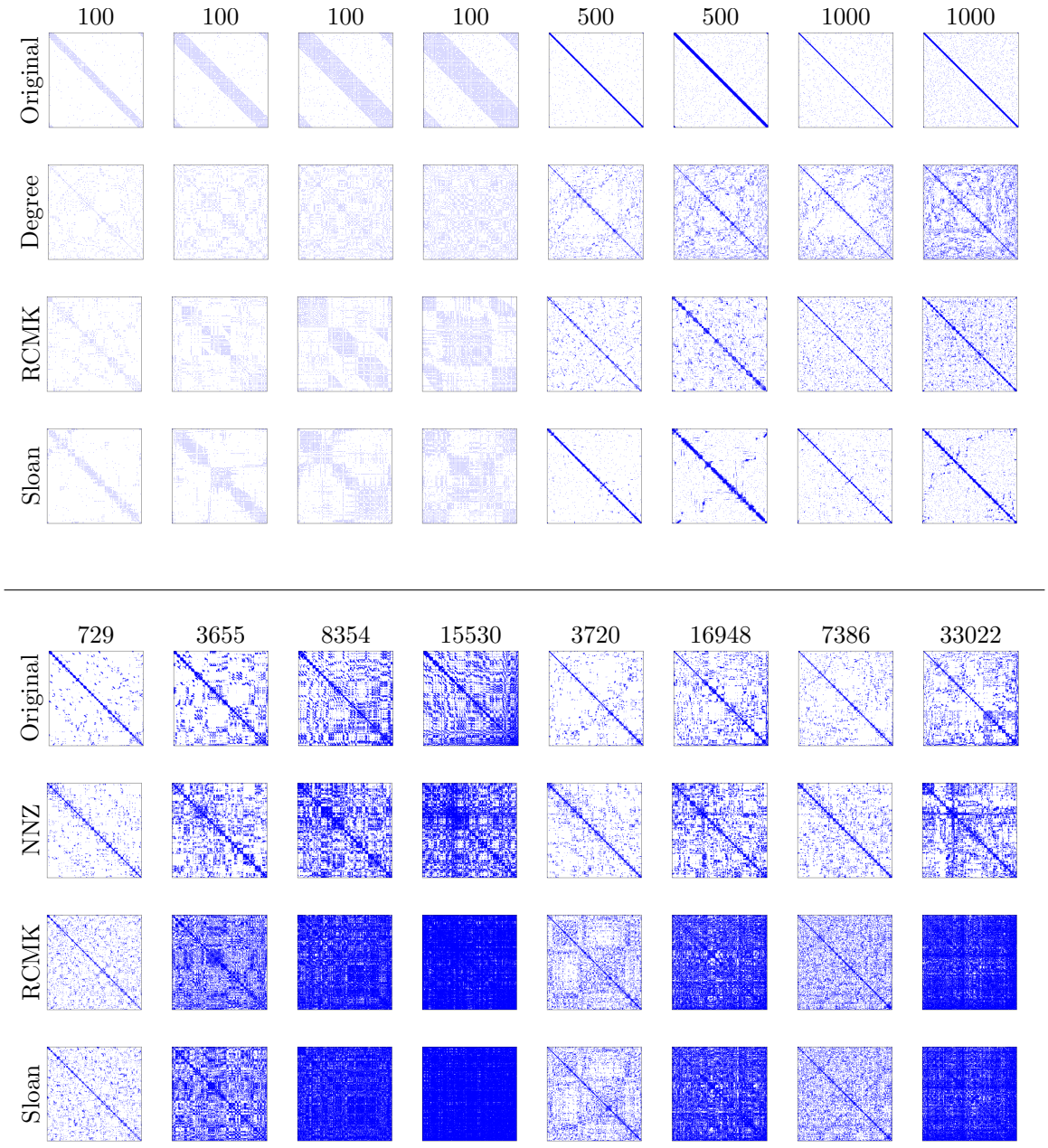


Figure 10: *Top four rows :* Δ_0 for Watts-Strogatz graphs. *Bottom four rows :* Δ_2 for Watts-Strogatz graphs.

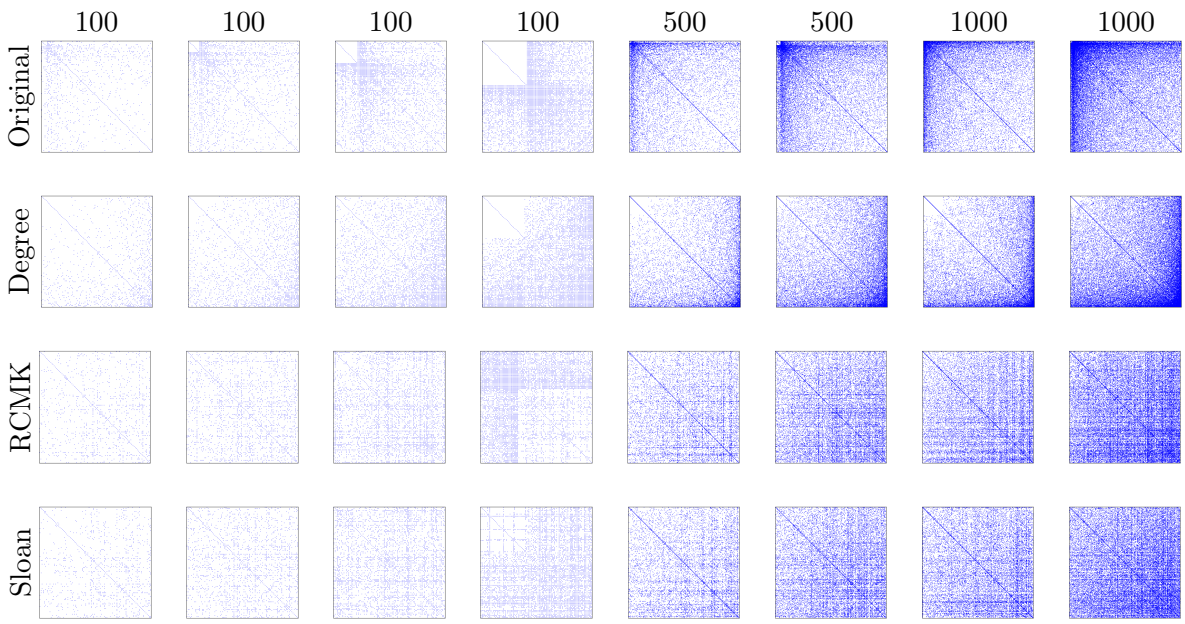


Figure 11: Δ_0 for Barabási-Albert graphs.

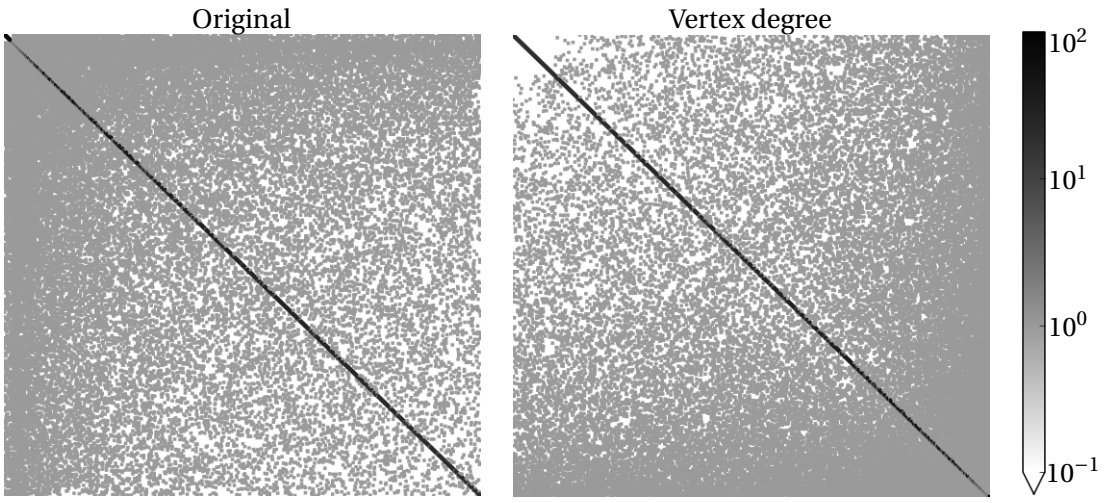


Figure 12: Enlarged view of the matrix corresponding to the last column in Figure 11.

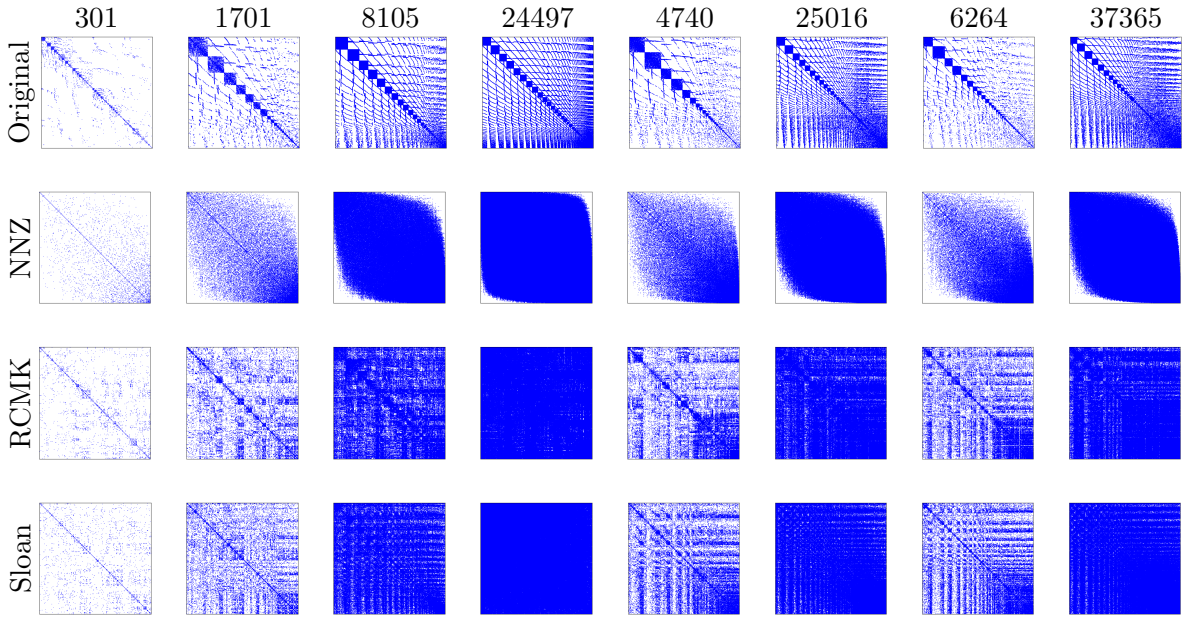


Figure 13: Δ_2 for Barabási-Albert graphs.

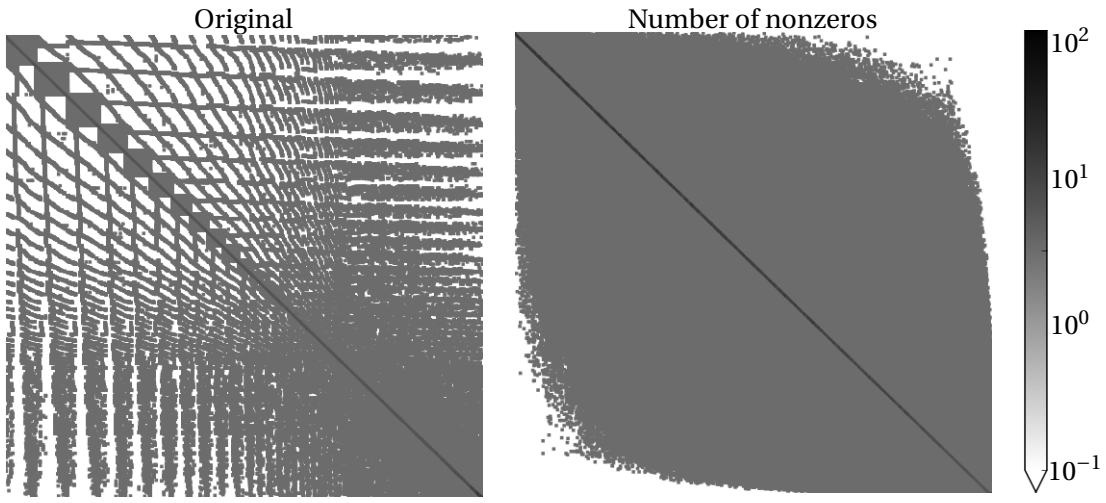


Figure 14: Enlarged view of the matrix corresponding to the last column in Figure 13.